João Luís Sacramento Salgueiro da Silva
Licenciatura em Engenharia Informática

A Solution for Estimates in Software Development Projects

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática

Orientador: Miguel Goulão, Professor Auxiliar, FCT/UNL
Co-orientador: João Quitério, Altran

Júri:

Presidente: Prof. Doutor(a) Susana Maria S. Nascimento
Arguente(s) Prof. Doutor(a) João Carlos Pascoal Faria
Vogal(ais): Prof. Doutor(a) Miguel Carlos Pacheco Afonso Goulão

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2014

João Luís Sacramento Salgueiro da Silva
Licenciatura em Engenharia Informática

A Solution for Estimates in Software Development Projects

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática

Orientador: Miguel Goulão, Professor Auxiliar, FCT/UNL
Co-orientador: João Quitério, Altran

Júri:

Presidente: Prof. Doutor(a) Susana Maria S. Nascimento
Arguente(s) Prof. Doutor(a) João Carlos Pascoal Faria
Vogal(ais): Prof. Doutor(a) Miguel Carlos Pacheco Afonso Goulão

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2014

Acknowledgments

I would like to express my gratitude towards my advisor Miguel Goulão, for the guidance, the dedication and all the effort spent throughout the elaboration of this dissertation. It was a magnificent experience and I learned a lot, as he could always find simpler ways to reach a solution, interesting material to read and showed a vast knowledge on uncountable things. I would also like to state that he was very professional, helpful and thoughtful. I consider him a role model and I can sincerely say that it was a pleasure to work under his supervision.
I would also like to thank my co-advisor João Quitério from for the availability, the help and the understanding the he showed on this period. His experience on the corporate world was enlightening for me and the participation on the brainstorming over this work was undoubtedly precious.
This dissertation was developed using information from the organization Altran Portugal, which enabled me to experience how a software development factory works and for this reason I am grateful to Maria da Luz Penedos, the person who represents the organization on this dissertation proposal and accepted my application to it.
A special thanks to my parents Luís Silva and Palmira Silva for advising me, for the concern they demonstrated and for being such a great example in life.
I want to thank my girlfriend Nance Sousa for the help, the availability, the concern and the support that she showed through my academic journey.
I am also grateful to my brother José Silva and my friends Paulo Ferreira and João Silva for the support and encouragement.
I would like to thank my colleagues at Altran Portugal for the experience they shared and for making it easy for me to adapt to the corporate world.
Finally, I would also want to express my gratitude for my family for giving me strength and for being one of the most important things I have in life.

Abstract

The Corporate world is becoming more and more competitive. This leads organisations to adapt to this reality, by adopting more efficient processes, which result in a decrease in cost as well as an increase of product quality.
One of these processes consists in making proposals to clients, which necessarily include a cost estimation of the project. This estimation is the main focus of this project. In particular, one of the goals is to evaluate which estimation models fit the Altran Portugal software factory the most, the organization where the fieldwork of this thesis will be carried out.
There is no broad agreement about which is the type of estimation model more suitable to be used in software projects. Concerning contexts where there is plenty of objective information available to be used as input to an estimation model, *model-based methods* usually yield better results than the *expert judgment*. However, what happens more frequently is not having this volume and quality of information, which has a negative impact in the *model-based methods* performance, favouring the usage of expert judgement.
In practice, most organisations use *expert judgment*, making themselves dependent on the expert. A common problem found is that the performance of the expert's estimation depends on his previous experience with identical projects. This means that when new types of projects arrive, the estimation will have an unpredictable accuracy. Moreover, different experts will make different estimates, based on their individual experience. As a result, the company will not directly attain a continuous growing knowledge about how the estimate should be carried.
Estimation models depend on the input information collected from previous projects, the size of the project database and the resources available. Altran currently does not store the input information from previous projects in a systematic way. It has a small project database and a team of experts. Our work is targeted to companies that operate in similar contexts.
We start by gathering information from the organisation in order to identify which estimation approaches can be applied considering the organization's context. A gap analysis is used to understand what type of information the company would have to collect so that other approaches would become available. Based on our assessment, in our opinion, *expert judgment* is the most adequate approach for Altran Portugal, in the current context.
We analysed past development and evolution projects from Altran Portugal and assessed their estimates. This resulted in the identification of common estimation deviations, errors, and patterns, which lead to the proposal of metrics to help estimators produce estimates leveraging past projects quantitative and qualitative information in a convenient way.
This dissertation aims to contribute to more realistic estimates, by identifying shortcomings in the current estimation process and supporting the self-improvement of the process, by gathering as much relevant information as possible from each finished project.

Keywords: Software Engineering Cost Estimation, Software Engineering Cost Prediction, Software Engineering Effort Estimation, Software Engineering Effort Prediction, Software Engineering Estimation Model

Resumo

No mundo empresarial existe cada vez mais concorrência. Isto leva as empresas a adaptarem-se ao ambiente competitivo, procurando adotar processos mais eficientes, que permitam minimizar os custos e que resultem em produtos de maior qualidade.
Um destes processos consiste na apresentação de propostas a clientes que incluem, necessariamente, uma estimativa dos custos do projeto. Esta estimativa é o foco principal do nosso trabalho.
Em particular, pretende-se avaliar que modelos de estimação são mais adequados à fábrica de software da Altran Portugal, empresa onde o trabalho de campo desta dissertação se realiza.
Não existe um consenso alargado sobre quais os tipos de modelos de estimação de custos mais adequados a projetos de software. Em ambientes ricos em informação objectiva disponível, os métodos de estimação baseados em *modelos formais* apresentaram melhores resultados que a avaliação de perito. Porém, o que acontece com maior frequência é não haver um volume e qualidade de informação significativo, com consequências negativas na performance de abordagens baseadas em modelos, favorecendo a utilização da *avaliação do perito*.
Na prática, as empresas usam, na sua grande maioria, *avaliação de perito*, o que faz com que fiquem dependentes do perito. Um problema recorrente é que a eficácia da estimativa desse perito depende da sua experiência em projetos idênticos. Deste modo, inferimos que o aparecimento de novos tipos de projeto leva a resultados imprevisíveis da eficácia da estimativa. Além disso, diferentes peritos tenderão a fazer estimativas inconsistentes entre si, com base na sua experiência individual. Como um todo, a organização não ganha, pelo menos diretamente, um crescente conhecimento sobre como a estimação deve ser feita.
Os *modelos de estimação* dependem da informação recolhida de projetos passados, no tamanho da base de dados de projetos e recursos disponíveis. A Altran atualmente não recolhe informação de input de forma sistemática. Tem uma base de dados de projetos pequena e uma equipa de peritos. O nosso trabalho é dirigido a empresas em contextos semelhantes.
Recolhemos informação da empresa, de modo a identificar que métodos de estimação se podem aplicar, considerando o contexto organizacional. Identificámos que tipo de informação extra a empresa necessitaria de recolher para poder usar outros métodos de estimação. Com base na nossa avaliação, na nossa opinião, a avaliação de peritos é a abordagem mais adequada para a Altran Portugal, no contexto actual.
Analisámos as estimativas realizadas em projectos de desenvolvimento e evolução da Altran Portugal. Isto permitiu identificar desvios de estimação comuns, erros, e padrões, o que motivou a proposta de métricas para ajudar os estimadores a produzir estimativas tirando melhor partido de informação quantitativa e qualitativa recolhida em projectos passados.
Esta dissertação visa contribuir para a construção de estimativas cada vez mais realistas, identificando pontos a melhorar no processo de estimativas atual e apoiando a auto-melhoria do processo ao longo do tempo, através da recolha de informação relevante de cada projeto.

Palavras chave: Custos de Estimação em Engenharia do Software, Previsão de Custos em Engenharia de Software, Estimação do Esforço em Engenharia do Software, Previsão do Esforço em Engenharia do Software, Modelos de Estimação de Engenharia de Software

Index

List of Figures

List of Tables

List of Abbreviations

SLIM: Soft lifecycle management
COCOMO: Constructive Cost Model
CART: Classification and Regression Trees
OSR: Optimized Set Reduction
OLSR: Ordinary Least Square Regression
ANOVA: Analysis of Variance
COBRA: Cost Estimation Benchmarking and Risk Analysis
KDSI: Thousands of delivered source instructions
KLOC: Thousands of Lines of Code
ISBSG: International Software Benchmarking Standards Group
MRE: Magnitude of Relative Error
MMRE: Mean Magnitude of Relative Error
CMMI: Capability Maturity Model Integration
OBIEE: Oracle Business Intelligence Enterprise Edition
BI: Business Intelligence
HR: Human Resources
VBA: Visual Basic for Applications
SFA: Sales force Automation
PM: Project management
AD: Analysis and Design
DEV: Development
TEST: Testing
PROD: Production
WBS: Work Breakdown Structure

1 Introduction

Software project management is an essential part of software engineering [1]. Project management is important because professional software engineering depends significantly on organizational, budget, and schedule constraints. A project can be defined as a temporary endeavor undertaken to develop a unique product or service [2]. A project manager is the person responsible for managing a project [2]. A project manager must ensure that the project meets and overcomes budget and schedule constraints, while delivering high-quality software. In spite of the variation from project to project, for most of them, important goals include delivering of the software in time to the customer, keeping overall costs within budget, meeting the customer's expectations and to maintain the development-team motivated. To increase the probability of achieving these goals, the project manager must make the most accurate resources estimation he can.

1.1 Motivation

According to McConnell [3], the primary purpose of software estimation is to determine if the targets of a project are realistic enough to allow it to be controlled to meet them.
The cost/effort estimation addressed in this work represents an assessment of the likely quantitative result of the number of labor units required to complete a project expressed in hours, days or weeks [2]. Accurate cost estimation is essential for budgeting a project, enabling the success of contract biding and helping to constrain (i.e. better control) the execution of the projects [4].
Unfortunately, as noted by Menzies and Hihn [5], cost over-runs are common in cost estimation. They often result from a spiral process, where budget underestimating leads to cost cutting, which in turn results in less quality assurance, verification and validation, ultimately leading to a lower quality of the project. Of course, managers can always include a safety margin in their estimates, to mitigate the risks of under-estimation, but doing so sacrifices the competitiveness in contract bidding. As such, costs estimation uncertainty is likely to increase a project's cost by a contingency over and above its normal profit. That means that the less cost estimation uncertainty, the closer the estimated cost comes to the real cost of the project. This allows the organization to become more competitive and to explore the market opportunity, making it possible to accept relatively low profit projects that may give the organization the opportunity to make a greater profit later, through the confidence relationship they can build up with the projects' promoters.
To reduce the estimation uncertainty, a project manager must study the company environment and determine how the estimation will be carried. There are several factors to consider, such as the size of the project, the type of software being developed, the personnel making up the project team, the programming languages used, and the used technologies, among other factors related to the project. All these are likely to influence the costs and schedule to be estimated.
Fairley [6] suggests three basic principles of estimation that managers should consider:

- An estimate is a prediction made from past experiences, adjusted accordingly to the differences between the current project and those conducted in the past.
- Estimates are made based on a set of assumptions that must be satisfied and a set of constraints that must be complied.
- Projects need to be re-estimated periodically as understanding increases and anytime project parameters change.

According to this set of principles, the project manager needs to have some experience on projects with equivalent characteristics, to be able to assess the differences between them and to adjust the estimate accordingly, depending on the requirements of the project at hand, and the changes observed in the context. The estimation process is often devalued because it consumes resources that could otherwise be devoted to the realization of the project. However, the time invested in estimation

represents but a small percentage of the time that would be spent on rework that occurs when this planning is not conducted [7].
In short, there is an important economic value in producing accurate estimates. An estimate is judged to be accurate, when the difference between the estimated and actual value is acceptable [8]. In order to increase the estimation accuracy, project managers can combine proven techniques with their own experience and the access to reliable historical data on previous projects. Estimates can, and should, be made in different moments in the software process. In general, the earlier an estimate is performed, the less accurate it will be. This condition can be verified with Boehm's cone of uncertainty (Figure 1)[9]. Early estimates happen when a company makes a bidding to win a project or whenever a client demands one. These estimates should always provide an error range representing the degree of confidence in the estimate. As the project progresses, there are increasingly less and smaller sources of uncertainty, leading to more accurate estimates.

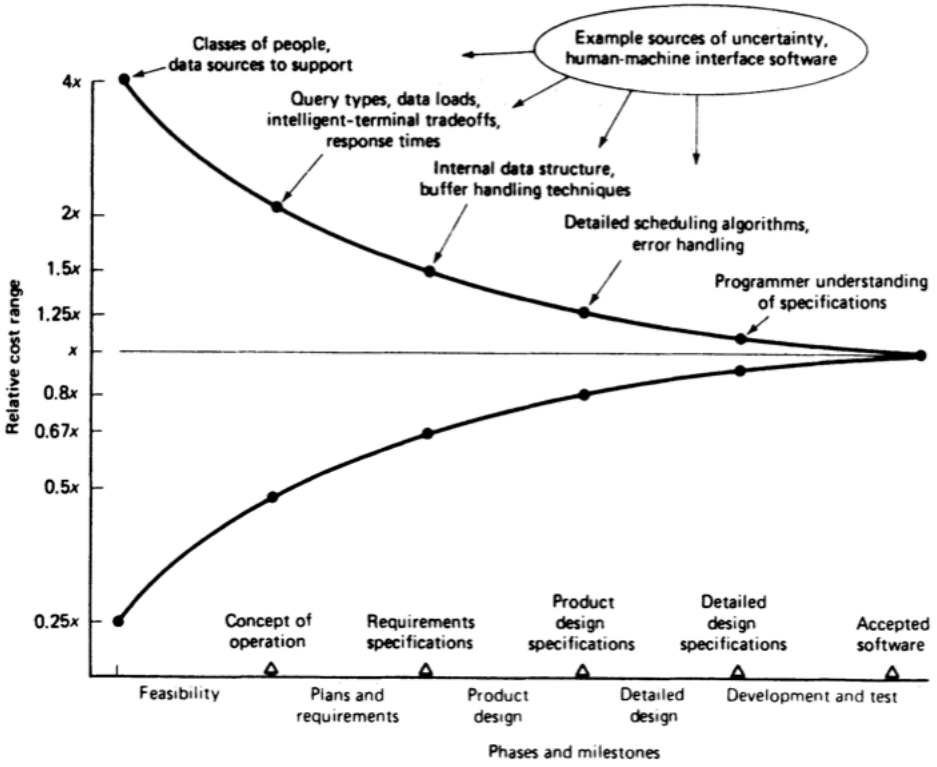

Figure 1 - Cone of uncertainty[9]

Accurate estimates have several benefits [3]:

- **Improved status visibility** – Comparing planned progress with actual progress is one of the best ways to track the project progress.
- **Higher quality** – Errors caused by placing stress on the development team can be avoided with accurate estimates**.** The development team tends to inject more errors on the project when under stress.
- **Better coordination with non-software functions** – Accurate estimates allow better coordination of the whole project, including both software and non-software activities.
- **Better budgeting** – The more accurate the estimate is, the more accurate the forecast of the budget will be.
- **Increased credibility for the development team** – A project team that insists on an accurate estimate will improve its credibility within its organization.
- **Early risk information** – When a risk is detected early, actions can be taken in order to minimize it, redefine the scope of the project or even decide that the project is not worth doing after all.

1.2 Context

As discussed in the previous section, the ability to consistently produce accurate estimates is instrumental to the competitiveness of organizations developing software. While this challenge is general, the specificities of each organization significantly constrain which kinds of estimation approaches are more suitable for those organizations. To the best of our knowledge, no single estimation approach is considered as "the best". So, each organization has to choose the approaches that best fit their own context.
This dissertation is carried out in the context of Altran Portugal, which is part of an international organization whose mission is to assist companies in their efforts to create and develop new products and services. In general, Altran Portugal uses the waterfall model and the majority of projects are web development. Like any other software producer organization, Altran Portugal needs to be able to produce estimates as accurately as possible, in order to bring forward better bids when competing for projects, as well as being able to assess with higher accuracy if a project is worth doing or not. The increased systematization on how estimates are performed is part of Altran's software process improvement initiative, which includes, among other objectives, a certification in CMMI [10], level 3.

1.3 Objectives

The main goal of this dissertation is to contribute to Altran's software process improvement initiative with respect to the way software estimates are performed. This involves identifying and assessing candidate estimation approaches in the context of Altran's software factory.
There are several candidate estimation approaches to consider available in the literature, with varying levels of competences required from the project managers conducting those estimations. While some approaches are model-based, others rely mostly in expert opinion. All approaches require their own specific sets of inputs, which involve varying levels of effort to gather. Of course, Altran already performs estimates and collects some data to support the currently used approach. Understanding which other alternative approaches could also be supported by the data currently collected, as well as which additional data would be required for enabling the usage of different estimation approaches is also an objective of this dissertation. To address this problem, we will assess estimation approaches in terms of a gap analysis concerning what input data they require, which part of it can be recovered from Altran Portugal projects data and which is already being recorded. To conduct this gap analysis, we are going to analyze all projects stored in Altran Portugal database in order to extract the ones that present sufficient data to be used as inputs to various candidate estimation approaches. We will also identify projects for which some relevant information is missing that can be recovered through meeting with the Project Manager allocated to that project. This will allow identifying which alternative estimation approaches could have been used, for the same project, as well as which extra information would have been required for enabling the usage of additional approaches, thus widening the set of alternatives.
Many estimation approaches rely on data from similar past projects to be successfully used. Estimators from Altran Portugal have access to previous project data. However, they can only extract one project at a time and do not have information on the aggregate type of projects. In other words, the estimator can only analyze one project at a time and does not have the full picture for similar projects. In order to suppress this disadvantage, we will group projects by their specific characteristics, analyze patterns useful to perform estimates and provide a framework to access these results on the behalf of estimators.
This way, estimators will check previous project data more easily, faster and it is an efficient way to acknowledge what typically happens in similar projects.
Altran Portugal has not only development projects, but also maintenance projects, where clients frequently submit change requests. Satisfying those requests also implies estimating the resources, time and effort necessary for their fulfillment. The accuracy of these estimates is relevant not only internally, but also for client satisfaction. As such, another goal of this dissertation is to evaluate an evolution project to detect improvement opportunities in it, supporting the organization's effort for continuous improvement and pursue for excellence on the services provided for its clients.

Table 1 summarizes our research questions and their motivation.

Research Question	Main Motivation
RQ: 1 – Which estimation approaches are applicable to the context of Altran Portugal, and suitable for the CMMI process improvement initiative?	This will allow us to identify the candidate approaches to be compared, excluding those available approaches that, for some reason (e.g. type of inputs required) are not a good match for Altran's specific context.
RQ: 2 – What are the estimation patterns observed in Altran Portugal projects repository and how can estimators use them to produce more accurate estimates?	This will allow us to identify where the deviations come from, how similar projects can be according to a chosen characteristic and to learn lessons from past projects to produce better estimates.
RQ: 3 – What can Altran Portugal improve in terms of client satisfaction concerning the evolution projects?	This will enable us to propose quantitative metrics that will enable a closer monitoring of the evolution process, leading to more accurate estimations in it which can ultimately leverage the quality of service as perceived by Altran Portugal's clients.

Table 1 - Research Questions

1.4 Expected Contributions

The expected contributions of this dissertation aim to address the research questions identified in the previous section.

The first important contribution is the identification of the candidate estimation approaches, including a gap analysis between the data Altran is currently collecting and the information needs of each of the considered estimation approaches. This implies conducting a survey on existing estimation approaches, analyzing the existing project repositories available at Altran Portugal, and mapping the information required by each approach to what is available currently in the organization's context. This mapping will allow identifying information needs that would enable using other estimation techniques than those currently in place at Altran Portugal.

The impact of this first contribution is that it will enable the selection of applicable estimation approaches, with the existing information, as well as establishing a more systematic approach to data collection, if additional estimation approaches are to be considered.

It is beyond the scope of this dissertation to make the decision on which approach(es) the company will adopt in the future, but this research is expected to help making an informed decision on this matter.

The second important contribution is to help estimators, giving them relevant past information on internal projects to enable them to make informed decisions when producing an estimate. We will be using historical data, not only because it enables the comparison of estimates with the actual values found later, but also to reduce the risk in live projects, making this research less intrusive in current projects.

The impact of this second contribution is that it will enable an estimator to have a higher variety of tools to perform the estimation and to obey to the CMMI level 3 estimation requirements, which includes the need of an estimator to use past project data to produce estimates.

The third contribution is closely tied to the second one. Rather than restricting only to internal projects data, we are also performing an assessment on evolution projects that are requested by Altran

Portugal's clients, this assessment will also consist in the analysis of the effort spent according to several project characteristics, however, it is also needed to propose metrics to measure client satisfaction and to understand what improvements are possible to make according to the results obtained.
Overall, Altran Portugal will benefit from this study, as the estimation approaches will be assessed taking into account the organization characteristics. This has the potential for helping Altran's estimation process to improve, both by helping to select a more accurate approach than the current one, and by increasing the awareness to the importance of identifying which additional data should be collected to help improving estimates with the current, or with other approaches. The assessment on the internal projects will also contribute to the awareness of what is currently happening on the organization, it will identify patterns useful to produce future estimates and it will enable us to learn from previous mistakes or good decisions. The assessment on the evolution projects is similar, however, it will bring more results concerning client satisfaction.
This awareness is important for Altran's software process improvement initiative. Altran's process currently meets CMMI Maturity Level 2. The outcome of this study may help improving the estimation process and have a positive impact in the process of achieving higher CMMI Maturity Levels, which require a more systematic, consistent and auditable approach to estimation that the one currently in place.
Other organizations, having similarities with Altran Portugal might as well retrieve positive outcomes from this study. If we succeed in the decrease of the effort estimation deviations, other organizations might choose to follow a similar approach, tailored to their own context.

1.5 Organization of the Document

Apart from this Introduction chapter, this document has the following structure:

- Chapter 2: this chapter consists in the analysis of the research of estimation models, the background and findings in the effort estimation and the actual state of practice.
- Chapter 3: here is presented work based on comparisons of estimation models and the use of internal versus external datasets.
- Chapter 4: we introduce the work plan and make the gap analysis.
- Chapter 5: we analyze the internal project database.
- Chapter 6: we analyze the evolution project dataset.
- Chapter 7: here is presented the conclusions and future work of this thesis.

2 Software Cost Estimation

In this chapter we discuss the state of the art in Software Cost Estimation.

2.1 Introduction

In order to answer the first research question (RQ1), presented in Table 1, we began by collecting relevant research papers concerning software cost estimation approaches. Jørgensen and Shepperd have observed that many important research papers concerning software cost estimation are difficult to find because there is lack of standardized terminology on this area [11]. Taking this into consideration, we searched for relevant research papers using keyword synonyms. After this, we made an assessment to identify the existing estimation approaches. Then, we identified the estimation models that belong to each approach, and proposed a scheme to classify the estimation approaches. The next step was to analyze each technique, and assess the characteristics we believed to be more important, in the comparison to be made between the techniques.

2.2 Estimation method Comparison Criteria

As we are conducting a study that aims the increase of the accuracy of the estimations, in the context of Altran projects, a comparison framework for Expert Judgment and Model Based Methods would be useful. The relevant dimensions we considered to compare the models are the accuracy of the produced estimates, the repeatability of the estimation model, the complexity that the model shows, the transparency of the process on which the estimation model relies on and the input required. In a systematic literature review on estimation approaches, Jørgensen, compares expert judgment and model based estimations [12]. This comparison shows that the accuracy depends strongly on the choice of the model and in the expert's experience. According to the evidence collected from the reported set of studies, the best experts outperform the best model. However, the best models outperform the average and least accurate experts. Furthermore, the least accurate models are still more accurate than the least accurate experts. This means that if an expert is new to some type of project, a model-based method may be a safer choice in terms of accuracy. Even with seasoned experts, having a model available to help them cross-checking their predictions, and investigating potential sources of deviation between their own estimates and those produced by the model is still helpful.

2.2.1 Accuracy

Accuracy is the measure of how close a result is to its correct value [13]. Once a project is finished, we can compute the estimation accuracy of the different approaches applied to it. To measure the accuracy of multiple estimation models, we have to aggregate a set of projects and apply the different estimation models to that set [14]. Sometimes, the results of the comparisons are erroneous because the data used in the comparison is different from model to model. Different models should be applied to the same set of data in order to compare the accuracy on that type of data. One model may outperform another in one type of projects, but the contrary can occur when applied to other type of projects. Accuracy can be obtained through several measures. The most used measure is the Mean Magnitude of Relative Error (MMRE). The Magnitude of Relative Error (MRE) can be obtained using the formula: $MRE_i = \frac{|ActualValue_i - PredictedValue_i|}{ActualValue_i}$, where i represents each observation to be predicted. The Mean Magnitude of Relative Error corresponds to the formula: $MMRE = \frac{1}{N_i} \frac{|ActualValue_i - PredictedValue_i|}{ActualValue_i}$, where N is the number of observations. In many studies, the Prediction level is also addressed (Pred (l)). The equation for Prediction level corresponds to:

$Pred\ (l) = \frac{k}{N}$, where k is the number of observations for which the MRE is lower or equal to l [15]. Another approach is the Balanced Relative Error (BRE). In [16], the authors state that BRE is a measure more balanced than MRE. The BRE equation consists in: $BRE = \frac{|Actual - Estimate|}{\min(Actual, Estimate)}$.

2.2.2 Repeatability

We consider the repeatability of a model, as the possibility of being able to easily reproduce the steps needed to apply the model and obtain the same output. This is important when estimating, because it makes the model more mechanical, leading to more consistent estimates for similar projects. When a model is difficult to reproduce, the estimator may obtain different estimate values for the same project. An estimator that uses Individual Expert Judgment based on Intuition on a project will probably not be able to reproduce the same estimation for the same project after a certain time. This leads to an inconsistency of estimates that other approaches try to attenuate.

2.2.3 Complexity

We address complexity as the difficulty of using and understanding an estimation model. In [15], the authors state that the more complex an estimation method, the higher the effort invested into estimates, the more error-prone, and the less likely it is for it to be adopted by practitioners. Altran Portugal has a team of project managers that currently use Individual Expert Judgment. Our goal is to improve the estimates. However, we also need to have a realistic approach when evaluating the estimation model, as we cannot expect project managers to adopt models that require too advanced statistical knowledge or rely on very complex functions.

2.2.4 Transparency

This is the characteristic that defines if a model is clearly explained, well documented and the estimation underlying process is visible to the estimator. If an estimator uses a model that hides the process that produces the estimation, when the model produces an inaccurate estimate, it is hard to understand what went wrong. As a result, project managers need to be able to view the process on which the model relies.

2.2.5 Type of information required

Different estimation models require different types of input. We need to assess the inputs on each estimation approach, in order to acknowledge what estimation models are applicable to the context of our study.

2.3 Estimation Approaches

Accurate software cost estimation has been a prevalent challenge for several decades and lead to the proposal of several estimation approaches and models. An estimation model is an unambiguous, reusable representation of the relationship between the effort, cost or productivity and its most important cost-drivers[15]. Cost-drivers are the variables that affect the cost of the estimates.
Several estimation models with a high impact in the software industry were proposed in the 1970s. In spite of the perception that there was significant room for improving their accuracy, relatively few new models have been developed since then [17]. This lack of novel approaches should not be interpreted as a symptom of a solved problem. As noted by the Chaos Report [18] there is still a large percentage of software projects that fail, or are challenged in terms of budget and time constraints, so more accurate software cost estimation can lead to relevant economical benefits for software producers.
Choosing the most adequate estimation model is far from trivial. As noted by Briand and Wieczorek, despite the various attempts to classify software estimation models, there is no agreement about which is the best [15]. As many other Software engineering decisions, the answer clearly depends on the context in which the estimation model is to be applied. Briand and Wieczorek propose a classification scheme for estimation models that we adapted and consider adequate to help in the comparison of

such models (Figure 2) shows the existing estimation methods in a hierarchy that distinguishes the various approaches that can be applied.

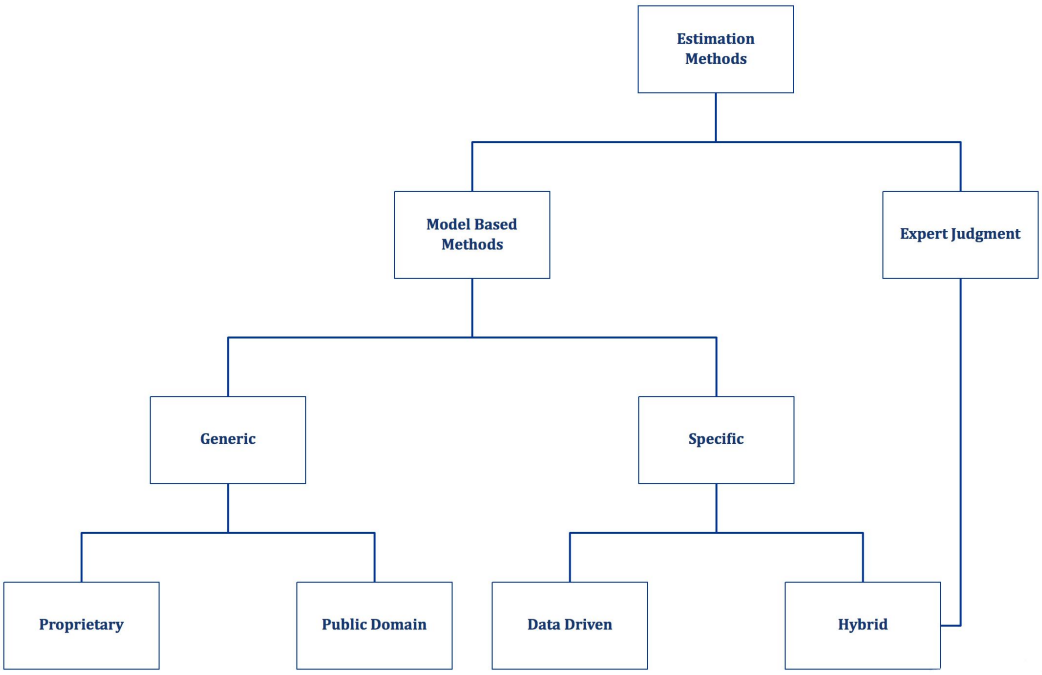

Figure 2 - Estimation Methods, based on[15]

A first important distinction can be made between Model Based Methods (the models used in these methods are often referred to as "Formal models") and Expert Judgment, as discussed in [12]. These two types of methods can be differentiated on the process applied in the method step where the estimation problem is quantified as a measure of the required effort to solve it. If the estimator does this process mentally, it is defined as Expert Judgment. Otherwise, if the process is performed mechanically, it is defined as model-based. Both model-based and expert judgment cost estimation, have several different variations.

Boehm identifies several benefits brought by software cost estimation models. These models help defining and prioritizing the features to include in the software product, how much hardware should be acquired to support the project and how much should be invested in tools and training. It also allows us to discuss how much money and time we should spend on the development of the software. In addition to this, a well-defined estimation model can help avoid misinterpretations, underestimates and over expectations. Unfortunately, Software Engineering decision analysis techniques are only as good as the input data we can provide for them [9]. This creates a challenge for organizations, as noted by Heemstra, because most software cost estimation models do not support project management in all the necessary steps, which should include [17]:

- Creation of a database of completed projects
- Size estimation
- Productivity estimation
- Phase distribution
- Sensitivity and risk analysis
- Validation
- Calibration

This insufficient coverage of project management is, then, one of the challenges to overcome when striving for using these estimation models.

From a structural perspective, most models are two-staged models. The first stage consists on estimating the size of the product to be developed. The most used size measures are *lines of code* and *function points*. *Lines of code* count the number of lines in the source code of the software application. *Function points* measure the amount of business functionality an information system offers to its users. The result of a sizing model is the size/volume of the software product to be developed, expressed as the number of lines of code or the number of function points. The second stage consists

on estimating the time and effort it will take to develop a software product of the size obtained in the first stage. First, the estimate of the size is converted into an estimate in nominal man-months of effort. A man-month corresponds to one person's working time for a month, or the equivalent, used as a measure of how much work is required or consumed to perform some task. Then, as this nominal effort does not take advantage of the knowledge concerning specific characteristics of the software product, the way it will be developed and the production means, a number of cost-drivers are added to the model. These cost-drivers have impact on the development and this impact must be estimated. This effect is known as the productivity adjustment factor. When this correction factor is applied to the nominal estimation of the effort, more realistic estimates can be achieved [17].

2.3.1 Expert Judgment

In the early stages of a project, model based methods can make poor estimates due to the vague or ambiguous input information available. At this phase, Expert Judgment is more likely to provide a better accuracy of the estimate, as the expert can have important domain knowledge not included in the models. Expert Judgment involves consulting one or more experts that provide estimates using their own methods and experience [19]. This method can be categorized into two types, individual or group Expert Judgment. Individual Expert Judgment consists in a single expert conducting the estimation task, whereas group Expert Judgment requires a group of experts working together, each one contributing to the making of the estimation.

Expert Judgment is the most used method in the industry. Unfortunately, it is often the case that the estimates are not good enough. In [20], Jørgensen identifies some reasons why experts do not achieve good estimates and provides estimation guidelines. These errors, occur due to several reasons, such as pressure to produce the evaluation, irrelevant and untrustworthy information used when estimating, not using documented data on already performed tasks and not addressing the uncertainty of the estimation. To avoid this type of errors, an estimator should not carry out the estimation process under the pressure of a prize or punishment as each of them can bias the results in a different direction. While prizes foster over-optimistic estimates, punishments lead to over-pessimistic estimates. A very common problem is the over-optimism of the estimator. Over-optimism can be reduced if the person estimating is not the one performing the task. In addition, if an estimator criticizes and evaluates his estimations systematically, the accuracy of the estimation can increase. This self-assessment counter-measure also works with the threat of over-pessimistic estimates, which are frequently associated with a conservative approach for mitigating risks in cost estimation.

Another factor that can bias the estimation is the information provided by the client. The expert needs to be able to detect if the information is useful, irrelevant or misleading. The use of historical data to help estimating the tasks can lead to fewer biases and reduce the subjectivity in expert estimation. Using tools, such as checklists to assist experts in their estimates can also reduce this subjectivity.

According to Boehm [9], Expert Judgment provides the ability to assess the representativeness of a software project, finding the most similar previous projects and making an informal analogy. He also states that Expert Judgment can benefit from its characteristic of promoting interactions. An expert can gather relevant information through interactions with the various stakeholders and perform a more reliable estimate. Expert Judgment has also the advantage of being able to adapt to exceptional circumstances. When a new type of project arrives, similar projects may not be available due to the technology or size of the project and the experts can be more flexible, using the knowledge they have to perform the estimation.

Individual Expert Judgment

For smaller organizations, financial and resource allocation restrictions may lead to the use of individual expert judgment [21]. This model requires only one expert to perform the estimate. It is also usually used in situations where a first indication of effort and time is needed, especially in the first phases of software development in which the specifications of the product are vague and continually adapted [17]. In [15], the authors state that when applying this model, experts use their experience and understanding of a new project and available information about the new and past projects to derive an estimate. Experts can use a lot of information to perform the estimates. The procedure to obtain the

final estimate can be more or less structured, depending on the individual. Lederer and Prasad [22] identified the basis of this estimating process:

- Analogy-based:
  - Comparison to similar, past projects based on personal memory
  - Comparison to similar, past projects based on documented facts
- Intuition
- A simple arithmetic formula (such as summing task durations)
- Guessing
- Established standards (such as averages, standard deviations, etc.)
- A software package for estimating
- A complex statistical formula (such as multiple regression, differential equations, etc.)

Some problems can arise when using this model. The reliability of the estimation depends highly on the experience and ability of the expert on the type of project he will estimate [17]. An inexperienced estimator on a certain type of projects is more likely to produce bad estimates. It is difficult for someone to reproduce the estimation made by another expert. The experience and knowledge of an expert cannot be simply passed to another estimator. The expert tends to be over-optimistic when he is to perform the estimated task himself [23]. Moreover, the organization becomes dependent on the expert. If the expert leaves the job, it is not possible to retain his experience and knowledge.

Individual expert judgment requires a single worker to perform the estimate, making it a cheaper solution in terms of resources and financial costs. As seen earlier, experts can use several types of information to derive estimates. This characteristic makes experts more flexible than models that need specific input information. Additionally, this approach takes advantage of the knowledge and experience of the estimator. The best experts generally perform better than the models [24][25].

Individual expert judgment does not specify which input information an expert needs to use. However, experts can use several information, such as design requirements, source code, software tools, rules of thumb, resources available, size/complexity of the new functions, data from past projects or feedback from past estimates [15]. This flexibility can also be a disadvantage, as this model is very difficult to repeat. Individual expert estimation can be complex or simple, depending on the basis of the estimation process. If the basis of the estimation relies on intuition, guessing, a simple arithmetic formula, a software package for estimating, established standards, or comparison to similar past projects based on documented facts, we consider it a low complexity model. By contrast, if the basis of the estimation relies on comparison to similar, past projects based on personal memory or a complex statistical formula, we consider it a high complexity model. In terms of transparency, this model is not transparent because despite using tools that help them conducting the estimating task, experts usually use their own process when estimating. The accuracy of this model highly depends on the experience and knowledge of the expert performing the estimation.

Group Review

Group review is a simple technique for improving the accuracy of the estimates that consists in a group of experts reviewing the estimate[3]. This model can be implemented by following three rules:

1. After making an estimate, the expert has each team member estimate pieces of the project individually, and then meets to compare with his estimate.
2. The estimator must not just average his estimates and accept that value.
3. Arrive at a consensus estimate that the whole group accepts.

When the estimates are compared, it is necessary to discuss differences in the estimates enough to understand why they are different and work until a consensus is reached on high and low ends of estimate ranges. Then, the estimator can calculate the average, but he also needs to discuss differences among individual results. Discussing differences is crucial to reach better estimates, as an estimator may change his estimate after a getting a different perspective provided by the other estimator. In fact, if an impasse is reached, the estimators must discuss differences until they come to an agreement.

However, group review may also be biased. People with stronger personalities may dominate the deliberations [26]. Besides that, group review may present some inherent disadvantages from individual expert judgment. On the other hand, group review can balance the over-optimism of some estimators with the caution of others. Taking this into account, the group of estimators should be

balanced to produce more accurate estimates. In spite of the estimators introduce bias if the over-pessimism compensates the over-optimism, the estimation becomes unbiased. The discussion of the estimates leads to sharing experience and consequently improving estimation skills. Experts learn from other experts that provide a better insight of some situations familiar to them.
Group review requires an individual expert judgment estimation of the whole project and one individual expert judgment estimation for each task in the project. In terms of repeatability, on one hand this process can be reproduced, but on the other hand it relies on a number of individual expert estimations that are not repeatable. This model shows very low complexity and also high transparency. Group-reviewed estimates have been reported to provide estimates with an average of 30% [3].

Wideband Delphi

This estimation model is a modification of the Delphi method created by Boehm and his colleagues [21]. To understand the Wideband Delphi model, we first discuss the Delphi estimation method. The Delphi method consists in a set of procedures for eliciting and refining the opinions of a group of experts or especially knowledgeable individuals [27]. The author states that the Delphi procedures were designed to reduce the undesirable effects of group interaction, such as socially dominant individuals and group pressure. The four key features of the Delphi technique are anonymity, iteration, controlled feedback and statistical aggregation [28]. Delphi achieves anonymity by using questionnaires, letting each individual answer privately. Besides that, iteration is achieved through the iteration of the questionnaire a number of rounds, so that participants can change their answers. The controlled feedback is achieved, because between iterations, the participants receive feedback from the other participants. Finally, statistical aggregation is obtained through the presentation of the final estimate as the statistical average of the individual estimates of each participant. Thus, there is no particular attempt to arrive at unanimity among the participants, which results in a spread of opinions on the final round [27]. An initial study on the Delphi method concluded that this technique was not as accurate as expected. Boehm and his colleagues found that Delphi meetings were subject to too much political pressure and likely to be dominated by the more assertive participants [3]. Considering this, they extended the Delphi technique and created a method called Wideband Delphi. Figure 3 shows the Wideband Delphi process flow.

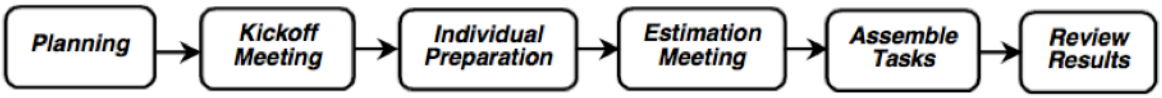

Figure 3 - Wideband Delphi process flow[29]

In [29], Wiegers describes the process flow of Wideband Delphi. The process starts with the planning phase. The person who initiated the estimation activity provides a specification of the problem to each participant. This specification must have enough information to produce credible, informed estimates. The participants include a moderator, the project manager, and two to four other estimators. Then, all participants attend a kickoff meeting. At this meeting, the moderator explains Wideband Delphi to the participants that are unfamiliar with it and provides the problem specification and any assumptions or project constraints. When the moderator concludes that all participants have sufficient knowledge, the group is ready to advance to the next phase. At the end of this phase, the following conditions must be satisfied:

- The team members are appropriate
- The kickoff meeting was performed
- The estimators agreed on estimation goals and units
- The estimators are able to participate effectively with the information provided

The individual preparation consists in the development of an initial list of the tasks that will have to be completed to reach the stated project goal, made by each participant using a form and an estimate for each task. The estimation guidelines for this phase are as follows:

- The expert assumes he will perform the task
- The expert assumes all tasks will be performed sequentially

- The expert assumes he can devote uninterrupted effort for each task
- In units of calendar time, the expert lists the expected waiting times you expect between tasks.

The next step is the estimation meeting. This meeting starts with the moderator collecting all the estimates performed by the participants and creating a chart with the final estimates. The moderator does not identify who created each estimate. After this, each participant reads his initial task list, identify the assumptions taken into account and raise any relevant issues or questions. The combination of the task lists will lead to a more complete list of tasks. After discussing and combining the tasks, each participant performs a new estimate. The moderator collects these estimates and plots them on the previous chart. The procedures of the estimation meeting are repeated until:

- Four rounds were completed
- The estimates converged to an acceptable range
- The allotted estimation meeting time is over; or
- All participants presented their final estimates

The next phase is to assemble tasks. The moderator or the project manager assembles the project tasks and the individual estimates of each estimator into a single task list. Individual lists of assumptions, activities and overhead tasks and waiting tasks are also merged. The final phase consists in reviewing the results. The participants review the summarized results and reach an agreement on the final outcome. In order to understand if the team is ready to close the estimation, the author also gives the following guidelines:

- The overall task list has been assembled
- A summarized list of estimating assumptions was created
- The participants reach consensus on how the final estimate was originated

The major drawbacks of Wideband Delphi are related to the practicality of the method. Firstly, obtaining the group opinions can consume too much time. Then, the number of experts required to apply this method can also be prohibitive [30]. Although there are downsides, this model also has some benefits. At first, this model helps creating a complete task list for major activities. Each participant needs to think of tasks that a specific activity and make a list, after that, the lists are discussed and merged to produce a complete task list. The creation of multiple estimates makes the participants acknowledge the uncertainty of the estimation. When the participants meet to get to an agreement, the anonymity of the owners of the estimates helps reducing bias. This way, an estimator avoids pressure from other participants or even being influenced by an estimation of a more experienced estimator. Additionally, a person is generally more committed to estimates he helps producing than to an estimate generated by others [29].

This model receives as input the task list proposed by the participants, the assumptions for the estimation, the estimates for each task and the list of waiting times the estimator expects to encounter. Regarding repeatability, this model has a repeatable process flow. However, the estimates provided by each participant are not repeatable. Although the process is repeatable, this model applied to the same project can present different estimates, as the participants may not be able to reproduce their estimates for each task. We consider that this model presents low complexity. In terms of transparency, Wideband Delphi is very transparent, because it is well documented and can be strictly followed. In [3], the author states that wideband Delphi improves estimation accuracy by an average of approximately 40% when compared to group averaging of individual expert judgment.

Planning Poker

Planning Poker is a model that was created by James Grenning. This model was created to eliminate situations when the estimation team takes too long to perform an estimate and when there are elements in the team that do not participate in the estimation [31]. To apply this model, the project must be decomposed in user stories, as it was created in an agile methodology environment. The Planning Poker model consists in a customer reading a story to be estimated to the programmers. The story is clarified, if necessary. Then, each programmer writes his estimate on a card and waits for the others. When all the programmers wrote their estimate, all cards are turned over. If the cards show an agreement in the estimates, the estimation is recorded and the next story is read. In case there is no agreement in the estimates, programmers discuss their estimates and try to get to consensus. If the team cannot reach a consensus, they can defer the story, split it or take the low estimate.

One of the disadvantages of Planning Poker can be the fact that a person with a stronger personality can interfere in the estimates. A person with a dominant personality is more likely to change the mind of another individual. Besides that, important discussions might not happen, as if the cards show an agreement, the team proceeds to another story. Another disadvantage is that discussions can end in very distinct estimations. When people do not reach a consensus after some time, the team moves to another story.
Concerning advantages, Planning Poker promotes the involvement of the whole team, as each programmer is obligated to play a card. It also can speed up the estimation process, because if the cards played show agreement, the group moves on to the next story [31]. Planning Poker also helps estimators avoid being influenced in their initial estimation, as the estimates are shown simultaneously. Another benefit is the combination of knowledge from different estimators [16].
This model takes as input a collection of user stories. The process itself is repeatable, but the estimates provided by each programmer are difficult to repeat. We believe this model presents very low complexity and is very transparent. In [16], Planning poker presented similar accuracy to individual expert judgment. Thus, the study concluded that both approaches had fairly unbiased estimates.

2.3.2 Model Based Methods

This type of Estimation methods usually takes a number of inputs and produces a cost estimation[15]. Some of those inputs, such as complexity and size of a program's module, are typically based on expert judgment because they are not known with high precision in an early stage of the estimation [12]. This means that model-based estimation may strongly depend on expert judgment-based input.
Model-based estimation leads to a better accuracy when calibrated to the situation in which the estimation models are used [12]. So, in general, before using a model, validation and calibration are necessary [17]. However, this calibration requires relevant input information from similar projects, which may not be readily available. Most of the times, the context on which the estimation model to be adopted was developed is different from the one of the project where the model is to be used. To make validation and calibration possible, the organization needs to have data on historical projects available. Many organizations do not have sufficient data on past projects. This has a negative impact on the potential success of estimation models built with data collected in projects within the company. An alternative, to counter this effect, is to use an external project database in order to gather relevant input data from similar projects built in other organizations.
A recent survey has shown that while some organizations could benefit from cross-company data, this is not necessarily the case [32]. Further research is necessary to fully understand this phenomenon, but existing evidence collected in the survey suggests that the type of projects and the characteristics of the organization might have to be considered when choosing an external dataset. This survey shows that the within-company datasets significantly outperformed the external datasets whenever the datasets were small and the cross validation was not very stringent. This means that the cross validation can possibly bias the results in favor of internal datasets. It was also observed that, all the studies where the datasets showed similar results had the within-company dataset as a subset of the cross-company dataset. Collecting data according to the cross-company data set can lead to not having a homogeneous group of projects. In the cases where a single company provided the data, the majority of studies presented better results for within-company data sets. This could be a consequence of the single companies being small organizations. Another factor that can contribute for the better results of within-company data is the relatively small datasets provided by the companies. This study does not cover companies that present lack of information on past projects, but we believe that using cross-company datasets can be valuable for our research.
There are several reasons why external project databases should be considered as a viable alternative to in-house data [32][15]:

- The time needed to accumulate enough data on past projects from a single company may be prohibitive.
- The time required to gather a dataset large enough to be used may make the data of older projects useless as technologies used by the organization may have change.
- It is necessary to have care, in order to collect data in a consistent manner.

Model-based methods have the advantage of reducing human or situational biases, the ability to weight variables more correctly than Expert Judgment and to produce consistent estimates [12].
Model based methods can either be generic or specific models. Generic can be applicable in different contexts [15]. These models are built using multi-organizational data [33]. These types of models assume that different relationships exist across environments. In order to apply generic models, the estimator has to investigate relationships based on data collection. This type of model requires the estimator to make an analysis in order to capture the most relevant cost-drivers in the context he is working on. Despite being generic, these methods require calibration to the context where they are to be applied in order not to present highly inaccurate results. As a result of using predefined cost-drivers, these models are not necessarily valid in every context. In addition, the predefined cost-drivers usually do not have clearly adequate or comprehensive definitions for a given environment. As a result, the estimator needs to come up with a different vocabulary and set of definitions, which is neither easy nor practical [15]. On the other hand, a study showed that specific models did not yield better results than generic models [34]. This type of estimation models also benefit from the use of multi-organizational databases. Multi-organizational databases offer larger and more up-to-date datasets when proper data collection is assured. The authors also stated that homogeneity within the projects of one organization might not be higher than across companies, as long as the projects belong to similar application domains and the data quality is high.
Generic models can be proprietary or public domain. Proprietary models are not fully documented or public domain [15]. These models are implemented as a black box, keeping the underlying information hidden.

PRICE-S

This model was created by an organization (RCA Corporation) for internal use on software projects [35]. However, at a later time, this model was released as a proprietary model, keeping the model equations away from public domain. Today, PRICE-S is marketed by PRICE Systems [36]. This model consists of three sub-models:

- Acquisition
- Sizing
- Life-cycle cost

The acquisition model predicts software costs and schedules. The Sizing model helps estimating the size of the software product at hand. The Life-cycle cost model is used to provide an early estimate of the maintenance and support phase of the software [35].
The model receives as input the project size estimate, the project application area, the level of new design and code, experience and skill of the team allocated to the project, hardware constraints, customer specification, reliability requirements and development environment. Moreover, PRICE-S presents low repeatability, as it relies on several inputs obtained using a mental process from the estimator. In terms of complexity, this model is seen as a "black box", which leaves us with no possibilities of addressing this characteristic. In terms of transparency, the model is not publicly available. Furthermore, we could not find information in the literature regarding the accuracy of this estimation model.

ESTIMACS

ESTIMACS focuses on the development phase of the software life cycle [35]. ESTIMACS identifies the dimensions of the estimation and correlates them to project factors. The model supports an iterative approach to develop final estimates. The iteration steps are, the input of data to feed the model, after this, an estimate is made and analyzed, and the results of the analysis are used to revise the input data. This model consists of five models:

- System development effort estimation
- Staffing and cost estimation
- Hardware configuration estimates
- Risk estimator
- Portfolio analyzer

The first model estimates the development effort as total effort hours through the answer of 25 questions related to the project organization and the system structure. The staffing and cost estimation takes as input the effort estimation and estimates the team size, staff distribution and cost. The hardware configuration estimates model provides the estimation of the operational resource requirements of the hardware to develop. The risk estimator estimates the risk of successfully completing the project at hand. Finally, the portfolio analyzer uses past projects to schedule the current project.
ESTIMACS receives a measure similar to function points and the answer to 25 questions to generate the model, needing tool support. The inputs may be divided in six groups: size variables, product variables, environment variables, personnel variables, project variables and user factors [15]. This model presents low repeatability, because, as PRICE-S, the inputs are derived from a mental process conducted by the estimator. In terms of complexity, this model cannot be defined, as the underlying information is hidden from the estimator. Regarding transparency, the model is not publicly available. In terms of accuracy, a study performed by Kemerer showed that this model outperformed SLIM and COCOMO [37]. However, the accuracy of ESTIMACS did not present satisfactory results, as its average error was 85%.
On the other hand, public domain models are fully documented and public domain [15]. These models provide information that enables the estimator to understand how the model works and which factors contribute to the accuracy of the estimation.

COCOMO – COnstructive COst MOdel

COCOMO is a model that predicts the effort and the duration of a project, based on input information regarding the size of the project and a number of cost-drivers that Boehm identified that affect productivity [37]. COCOMO I includes a set of three modeling levels: Basic, Intermediate and Detailed [15]. These modeling levels include a relationship between system size and development effort. The sizing is measured according to the delivered system instructions produced and the effort is measured in man-months. Basic COCOMO uses the formula $ManMonth = a\ Size^b$ to express the relationship between effort and size. We will describe the different COCOMO modeling levels, based on [19]. Basic COCOMO uses the coefficients a and b to calibrate its value to express the complexity of the software as follows:

- For simple, well-understood applications, a = 2.4, b = 1.05
- For more complex systems, a = 3.0, b = 1.15
- For embedded systems, a = 3.6, b = 1.20

Intermediate and Detailed COCOMO adjust the basic COCOMO formula, in order to account additional cost-drivers. There are 15 ranked cost-drivers that increase or decrease the nominal effort [15]. These models use the formula $Effort = a\ Size^b\ \prod_{i=1}^{15} EM_i$, where EM_i is a multiplier for cost-driver i. Intermediate COCOMO can be applied when the major components of the product are identified. Detailed COCOMO works on each sub-system separately and uses cost-driver multipliers that differ for each development phase.
The negative aspects of COCOMO I, are that it is hard to estimate KDSI (Thousands of delivered source instructions) accurately in an early stage of the project. In addition, as many cost-drivers are not considered, the basic model provides a rough estimate. The positive aspects of COCOMO I are that the estimation can be updated during the different stages of development, using the basic model at the beginning, the intermediate when the major components of the system are known and the detailed model when it is possible to identify each task to develop. COCOMO I also helps the estimator to understand the impact of the different factors through the cost-drivers.
COCOMO I model takes as input an estimated size of the project in KDSI and a set of 15 cost-driver values (the second input can be discarded if operating the basic modeling level). The repeatability of the process is high, but the size received as input is obtained by expert judgment and may not be repeatable. In terms of complexity, this model has low complexity. As the model is one of the best-documented [15], it has a very high level of transparency. COCOMO has been reported to provide low accuracy, when not calibrated to the context it was used in [37][24].
The shortcomings of COCOMO I, along with the evolution of software development approaches, lead to the development of an improved version, known as COCOMO II. COCOMO II also includes a set

of three models: Applications Composition, Early Design and Post Architecture [15]. Application composition involves prototyping efforts. The early design uses a few cost drivers, because at this stage, the specifications are still vague. The post architecture is usually applied when the after the software architecture is well defined. It is a detailed extension of the early design model, provides an estimate for the entire development life cycle, it uses 17 cost-driver multipliers and 5 exponential scale factors to adjust for project. The COCOMO II formula is equal to the COCOMO I detailed and intermediate, but instead of having 15 cost-drivers, it has 17.
The COCOMO II model takes as input a size estimation of the project in KLOC or Function Points and a set of 17 cost-driver values (as in COCOMO I, the second input is not needed to apply the first modeling level). The repeatability of the process is high. However, the size estimation of the project is obtained by Expert Judgment and may also not be repeatable. The complexity of this model is low, as it does not require estimators to make a great effort to understand the model. Regarding transparency, this model is also well documented, providing a high level of transparency for the estimator. In terms of accuracy, COCOMO II presents several versions to improve this characteristic, however, comparisons with other estimation models were not found.

SLIM – Soft Life-Cycle Management

SLIM is an estimation model created by Putnam, based on an equation of staffing profiles for research and development projects [15]. This model assumes that the Rayleigh curve can be used to model staff levels on large software projects. It supports most of the popular size estimating methods and uses the Rayleigh curve to estimate project effort, schedule and defect rate [35]. The MBI (Manpower Buildup Index) and a Technology Constant or the PF (Productivity Factors) influence the shape of the curve. SLIM can record and analyze previously completed projects to achieve calibration. If data is not available, a set of questions can be answered to get values of the MBI and PF from the SLIM model database. SLIM uses productivity to link the basic Rayleigh manpower distribution model to the size and technology factors of the project. An example of the Rayleigh curve can be seen in Figure 4.

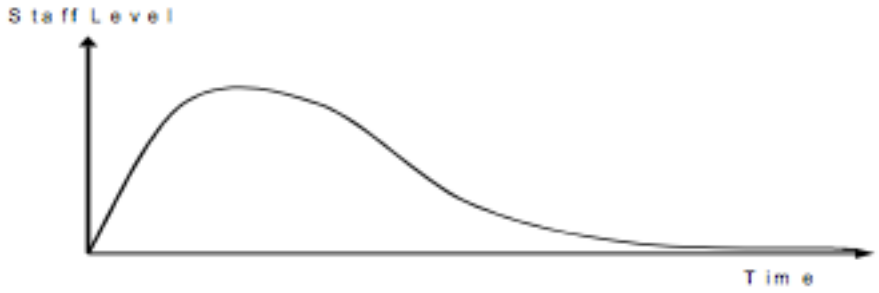

Figure 4 - Rayleigh curve example[15]

The equation that expresses the size of the project is called the software equation and is expressed as: $S = PF \times (Effort)^{1/3} \times t_d^{4/3}$, where t_d is the software delivery time and PF is the productivity factor [19]. The equation used to get the Effort is expressed as $Effort = D_0 \times t^3$, where D_0 is the MBI. The size is measured in KDSI and the Effort is measured in man-years. Combining the two equations gives us the power function form: $Effort = (D_0^{\frac{4}{7}} \times PF^{-9/7}) \times S^{9/7}$ and $t_d = (D_0^{-\frac{1}{7}} \times PF^{-3/7}) \times$.
There are already SLIM tools available, however these tools are proprietary. The model is also not suitable for small projects and the requirements analysis and feasibility studies are excluded from the estimation [15]. On the other hand, if the organization does not possess data on previously finished projects, SLIM provides a set of questions that an estimator can answer to get the MBI and PF. This set of questions enables the model to get the values of the MBI and PF based on a dataset SLIM possesses. It is also easy to calibrate this model, as the organization only needs to provide the size, effort and duration of past projects.

This model receives the MBI, PF and an estimate of the system size as input based on KDSI, or a set of answers to a questionnaire. The repeatability of the model is low, as its input is generated through individual expert judgment. In terms of complexity, the model shows medium complexity, as the estimator needs to understand a set of mathematical equations. SLIM shows high transparency, as the estimator is able to observe how the model works internally. Concerning accuracy, in study [37], despite presenting low accuracy, SLIM outperformed COCOMO I.
Specific Model Based methods can be data-driven or hybrid models. This type of model has its validity only ensured in the context where it was developed [15]. These models are usually calibrated through within company datasets and can only be applied by organizations that present characteristics according to the specifications of the model.
Data-driven models have data analysis as input information and can be distinguished as parametric or non-parametric. Parametric methods require that a functional relationship between cost and project attributes be specified. On the contrary, non-parametric methods do not require functional relationships to be specified [15].

CART – Classification and Regression Trees

CART makes use of two types of decision trees, classification and regression trees. Classification trees are used to make predictions for a variable quantified by a categorical value [15]. Regression trees are used to make predictions along a continuous interval or ratio scale and classify software projects according to a variable. A regression tree is a collection of rules and forms a stepwise partition of a dataset. Each node of the tree specifies a condition based on the selected project variable and each branch of the tree corresponds to possible values of this variable.
In order to build a regression tree, the dataset needs to be split recursively until a stopping criteria is satisfied [34]. In [15], Briand and Wieczorek present three key elements to build a regression tree:

1. Effectively split each node in the tree recursively
2. Decide when a tree is complete
3. Compute relevant statistics for each leaf

As CART uses decision trees, it inherits the drawbacks of this kind of trees. In the first place, the tree becomes too large when trying to express concepts hard to learn. If an estimator does not have enough experience building regression trees, he might create too complex trees. On the other hand, decision trees are easy to understand and they also provide fast computation. Another positive aspect of CART is that it can be adjusted in time. When new projects finish, they can be used to feed the model, resulting in more robust results.
This model receives as input a project dataset and the variable over which the projects will be classified. This model has high repeatability, as it can be easily reproduced. This model also shows high complexity, as an estimator needs to know how to build a regression tree. In addition, CART has high transparency, allowing the estimator to trace how the estimate reached a certain value. In terms of accuracy, in [34], CART performed a little better when compared to stepwise ANOVA, OLS and analogy based methods.

OSR - Optimized Set Reduction

OSR was created to determine which subsets of a project dataset provide the best characterization of the project to be assessed [38]. It consists on a set of logical expressions that represent trends in a dataset that are relevant to the estimation that is being conducted [15]. The model provides a different subset of similar projects for each new project to estimate. This is a stepwise process that chooses an independent variable at each step, reducing the subset and retaining the similar projects until a stopping criterion is reached. The estimate is based on a terminal subset that optimally characterizes the projects to be estimated. An optimal subset is characterized by a set of conditions that are true for all projects in that subset and they have optimal probability distributions on the range of the dependent variable. In practice, they concentrate a large number of projects in a small number of dependent variable categories or on a small part of the range. The model is also able to use several subsets to generate a range of predictions that reflect the uncertainty of the estimation.
The drawbacks of this model consist in the fact that the database needs to have sufficient projects to feed the model and if the estimator characterizes the project in a wrong way, the model may create a

sub set of projects with different characteristics, leading to inaccurate estimate values. On the other hand, OSR uses the project database efficiently, as it is able to generate sub sets according to the characteristics of the project. OSR also integrates statistical and machine-learning approaches to empirical modeling and provides support for dealing with both partial information and model interpretation [39].
This model receives as input the characteristics of a project and a project database. It has a process that presents high repeatability, as the estimator can obtain the same estimation if the database is unaltered and the same characteristics of the project are chosen. In terms of complexity, the model presents high complexity because it cannot be developed manually and uses complex algorithms. Although OSR presents high transparency as the model is explained in detail and well documented. In terms of accuracy, according to study [38], OSR outperformed COCOMO I calibrated to the environment. When the authors removed outlier projects, OSR presented 50% MRE.

Stepwise ANOVA – Stepwise Analysis of Variance

Stepwise ANOVA is the combination of ANOVA and OLS regression [15]. ANOVA usually decides if the different levels of an independent variable affect the dependent variable. In case this happens, the variable has a significant impact. The levels of a variable are the values that these variables can take. The stepwise procedure applies ANOVA using each independent variable in turn, resulting in the identification of the most significant variables and removing its effect by computing the difference between the actual and predicted values. To obtain the impact of ratio and interval variables, stepwise uses OLS regression. This model results in an equation with the most significant factors.
This model is very difficult to interpret for non-statisticians. Although, stepwise ANOVA can deal with variables of different types and identifies independent variables.
This model requires a project database and a set of variables as input. It shows high repeatability, as it is possible to get to the final equation for the same input variables and dataset. In terms of complexity, the model shows medium complexity, as the estimator can implement this model manually, however, it is difficult to interpret. The model has medium transparency because it is not as well documented as the other models. According to study [34], stepwise ANOVA performed better than models using analogy, being slightly outperformed by CART.

OLS – Ordinary Least Square Regression

OLS regression assumes the establishment of relationships between one dependent variable (e.g. effort) to one or more independent variables (e.g. cost-drivers) [15]. Least squares regression needs to begin with the specification of the form of relationship between the variables. Then, the model fits the data to that specification in order to minimize the overall sum of squared errors. Furthermore, OLS depends on the homoscedasticity assumption. This assumption consists in the difference between the actual value and the predicted value does not change for projects.
This model has the disadvantage of being sensitive to outlying observations in the dataset it uses, which may cause misleading prediction equations. In addition, this model can only use interval or ratio variables. Moreover it relies on a large dataset and it is difficult to interpret by estimators without statistics background knowledge. On the other hand, OLS is a standard method and has been widely applied.
This model requires a project dataset as input and the establishment of relations between variables. It is a highly repeatable model, as long as the estimator uses the same input. In terms of complexity, the model shows medium complexity, as it can easily be implemented, but it requires some effort to interpret. Regarding transparency, the model is also highly transparent as it is well documented. In terms of accuracy, in [39] OLS regression outperformed CART, stepwise ANOVA, and analogy-based techniques.

2.3.3 Composite Methods

Composite methods are based on a combination of expert judgment and data-driven models [15].

COBRA – Cost Estimation Benchmarking and Risk Analysis

In [40], the authors state that algorithmic and parametric models are not extensively used in organizations. The reasons why these models are not considered constitute the motivations that lead to the creation of COBRA. Firstly, many organizations do not collect sufficient data to allow the construction of such models [17]. Moreover, many measurement programs showed to be inefficient, leading to a high mortality [41].

COBRA consists in the development of a productivity estimation model that can be divided in two parts, the cost overhead estimation model and the productivity estimation. The cost overhead estimate is produced using expert judgment, whereas the productivity estimate is supported by past project data. Cost overhead is the cost derived from several factors, which affect the cost of the project, resulting in its increase or decrease. To produce the cost overhead estimate, the project manager answers a project data questionnaire related to the project characteristics. Productivity is an average measure of the efficiency of production. The authors believe productivity is strongly related to cost overhead and provide an equation to calculate productivity: $P = \beta_0 - (\beta_1 \times CO)$, where P is productivity, β_0 is the productivity of an optimal project (i.e. a project that presents the highest productivity possible), β_1 is the slope between CO and P and CO represents the cost overhead.

The main limitations of this model are related to the expert judgment. COBRA requires experts to be available to fill the project data questionnaires and the cost overhead estimate depends heavily in the knowledge and experience of the estimator.

COBRA brings several benefits, such as the reusable cost overhead estimation model, for similar projects the estimate can be recovered, as it does not depend on the size of the project. Moreover, this model can be supported on a small project dataset, which is a major advantage for organizations that do not possess a large project database.

This model receives a project data questionnaire and a measure of the size of the project. In terms of repeatability, COBRA presents medium repeatability, as the cost overhead estimate depends on expert judgment, and different experts are likely to achieve different results, as it depends on experience and knowledge. However, if the same team of experts provides the cost overhead estimate, it is possible to achieve relatively similar results. In terms of complexity, this model presents medium complexity, as the steps needed to perform the estimates require some effort to understand. This model also presents a high level of transparency, as it is well documented. The accuracy this model presents is very high, in [40], the authors indicate that estimates produced with this model deviate, on average, 9% from the actual values. A variation of COBRA [42] also indicated good accuracy results, presenting a deviation of 14% from the actual values.

3 Related Work

3.1 Introduction

While describing the estimation approaches, we referred several case studies on their performance. In this chapter, we discuss studies that, as in this dissertation, aim to compare the different estimation approaches and the effect of using in-house versus cross-company data in those approaches.

3.2 Estimation Approaches Evaluation

Empirical evaluation of estimation approaches in industrial settings remains challenging. As noted by Jørgensen [11], there are relatively few estimation case studies conducted in industry. This is unfortunate, as there may be much to learn from these real-life case studies. The scenario is not better for controlled experiments. Juristo and Dieste report a low number of experiments in industry and argue that this results from the perception that conducting experiments is time-consuming for the project, causing delays and hence being rejected [43]. Moreover, in many organizations it is difficult to motivate experiments, as they are concerned about financial issues. In addition, it is difficult to do a pre-planning of all the details of the experiments in an agitated reality that is to be expected in an industrial setting. Last, but not the least, the term "experiment" itself turns out to be demotivating for the industrial partner, as it appears to convey a stronger focus on the academic rather than the industrial benefit that may result from those experiments. This dissertation contributes with an estimation case study conducted in industry, using real project data, thus contributing to the body of existing studies.

In [34], the authors compare OLS regression, stepwise ANOVA, CART, Analogy-based estimation, and combinations of CART with OLS and Analogy. In order to compare the models, the dataset used to feed the models was divided into multiple training and testing sets, calculating the accuracy for each one. The projects in the dataset were provided by 16 organizations. However, to determine the accuracy and the benefits of estimation models using within company data, they used a subset of 63 projects from a single organization. After applying the estimation models to the dataset, the techniques not involving analogy outperformed the ones using analogy, with CART yielding the best results (0.569 MMRE). The authors state that a reason why this might have happened is that the similarity function (to find similar projects), on which the analogy model is based, shows variables with equal influence on the selection of the most similar projects. Overall, the results also showed that simpler techniques, such as CART, perform at least as well as more complex techniques, which suggests that to achieve accurate estimates, the quality and adequacy of the data collection is the key solution, rather than the model. The author also states that instead of having a list of generic cost factors, organizations should devise their own important cost factors to achieve acceptable MRE. Moreover, a replication of this study was conducted [33]. It is important to replicate studies, in order to establish the validity and generalizability of the results. However, despite comparing the same estimation models, this time the dataset used was different, relying on the information provided by a variety of European organizations and domains. To determine the accuracy of estimation models using within-company data, 29 projects were extracted from a single organization. After applying each estimation technique, the authors observed that OLS regression and ANOVA_e (using effort as the dependent variable) performed significantly better than the other techniques. As stepwise ANOVA presents a significantly complex automation and it seems to perform as well as OLS, it is doubtful that any benefit can come with the use of stepwise ANOVA in a similar context. The results on this experiment reinforce the conclusions of the replicated study, as analogy does not seem to bring any significant advantage over other estimation techniques. The authors observed extreme outlier predictions with analogy-based estimation. It was found that, despite retrieving a perfectly matched analogue with respect to some cost factors, the projects greatly differed in the system size, which lead to a poor performance of these

models. This supports the observation on the previous study that the poor performance of analogy-based methods can be attributed to the equal weighting of the variables of the similarity function. Another conclusion is that the combination of techniques did not lead to any significantly improved estimates. Moreover, the authors stated that even the best models are not very accurate (0.32 MMRE).
Kemerer conducted a study to compare the estimation models SLIM, COCOMO, Function Points and ESTIMACS [37]. This study was performed using medium to large projects from a single company database. After filtering the desired projects, there were 15 projects to feed the models. The estimates of the models presented very low accuracy, in general, which lead to the conclusion that models developed in different environments, do not perform very well when not calibrated. Average error rates calculated ranged from 85% to 772%. Kemerer states that this variation can occur, due to the degree to which the productivity of the environments where the models were developed matches the productivity of the target environment. This study showed that ESTIMACS and Function Points yielded better results than COCOMO and SLIM. When trying to answer if the proprietary methods yield better results than public domain models, the results were inconclusive, as SLIM (proprietary) outperformed COCOMO (public domain) and Function Points (public domain) outperformed ESTIMACS (proprietary).

3.3 Within Company versus Cross Company Datasets for Estimation

The existing evidence concerning the real benefits of using data from internal projects as input to estimation approaches, rather than cross-company data shows conflicting results. After a thorough literature review, Kitchenham [32] was unable to come to a conclusion whether cross-company models should be used or not, concluding that further research on this area is required.
Jeffrey et al. [34] also make a comparison of models using single organization data versus multi-organizational data. The authors state that the external dataset presents similar types of projects and shows similar distributions in terms of application domains and target platform. When comparing data from a single company to the data provided by the multiple organizations, the authors realize that the estimation models presented similar results, showing statistically insignificant differences. The conclusion from this analysis is that there appears to be no advantage in developing company-specific estimation models using generic cost factors and sizing measures. An explanation for the similar accuracy results might be that the projects on the external dataset are very similar to the within company dataset in terms of application domain and target platforms. Overall, the authors state that to really benefit from collecting organization-specific cost data, organizations should investigate the important factors in the organization to be considered to design a tailored, specific measurement program. Another implication from the results observed is that it is possible for organizations using extern databases, to yield results as good as the ones using internal databases. Moreover, this study was replicated (as seen in 3.2) to verify if the conclusions the authors have reached remain valid [33]. The accuracy of the models using a within-company dataset, were not significantly better, which is consistent with the previous study observations. The authors state that there is no obvious advantage to use within-company datasets, when generic cost-drivers are collected. They also state that it is possible that the advantage of using a more homogeneous dataset is offset by a significantly smaller project sample available to feed the models. This replication supports the previous study, as the authors achieved consistent findings.
Jeffrey et al. [44] oppose OLS and Analogy-based estimation models, to compare the use of multi-organizational data using the ISBSG database with the use of within-organization data using a single company database. The ISBSG is an organization that establishes, grows, maintains and exploits repositories of software metrics to help improve the management of IT globally [45].
In order to create a more homogeneous cross-company dataset, the authors selected only projects that measured resources on the same basis as the internal dataset, projects with entries for system size and team size, and that were new projects. After performing the estimates on the within-company data set, the estimation models presented no significant differences in terms of accuracy. The estimates concerning the cross-company dataset revealed that OLS performed significantly better than the Analogy-based methods. The authors observed that some analogy-based models had significant differences when used on multi-company data. They concluded that in that context, the size

adjustment does not significantly improve the accuracy of the estimates. However, it was observed that the use of more than one analogue is a driving factor of significant accuracy improvement. The authors state that the results obtained are a reflection of the non-linear relationship between system size and effort within the ISBSG dataset and the wide range of these project characteristics. Moreover, when comparing the accuracy of the models on each dataset, when the models used the within-company dataset, they reached higher accuracy. It was also noted that internal projects had higher productivity values, which may explain the differences in the accuracy. The fact that OLS regression presented a smaller variance on the accuracy of the estimates, we can conclude that it is more robust than techniques relying on Analogy.

3.4 Discussion

Taking into account the results observed in each study, it is possible to detect some patterns. The way project data is collected to build data-driven cost models has to be properly addressed to achieve better estimates. Moreover, we can also state that simpler estimation models like CART and OLS provide at least, as good estimates as more complex estimation models. Analogy-based techniques yielded poor accuracy, as they are highly dependent on the similarity function to find better analogue projects, and it is not trivial to calibrate it. We can also conclude that, estimations using a cross-company dataset are dependent on the homogeneity of the projects and on their similarity to the target context. Furthermore, the combination of techniques might not be a plausible solution, as they yielded poor estimates on the observed studies.
The use of an external project database is not a case we are going to study because the quality of the information on the dataset we had access to was dubious.

4 Gap Analysis

4.1 Introduction

In this chapter we begin to present our proposed solution in terms of the set of activities identified in Figure 5. We began by identifying candidate estimation approaches. After this assessment, we conducted a gap analysis, in order to identify the estimation approaches that can be applicable to Altran's context. In our preliminary assessment of the existing project information, we detected some information missing from the project repositories. In order to increase the available estimation approaches, we performed an assessment on the internal project dataset and checked what missing information would be needed to perform estimates using other estimation approaches. This will bring us to a conclusion regarding the research question RQ1.
After the assessment on estimation approaches, we analyzed how Altran Portugal is currently estimating, using its internal project database. This assessment is detailed in Chapter 5.
Finally, we also studied the quality of the service provided by Altran Portugal to its clients, in the context of a software evolution project that includes a few hundreds of requests for evolution.

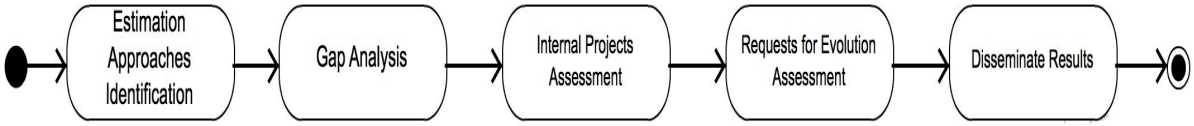

Figure 5 –Work plan

4.2 Estimation Approaches comparison

The results of model comparison are represented in Table 2. This table uses the characteristics presented earlier to compare the models.

	Inputs	Repeatability	Complexity	Transparency	Accuracy
Individual EJ	☺	☹	☺	😐	😐
Group Review	☺	☹	☺	☺	😐
Wideband Delphi	☺	😐	☺	☺	☺
Planning Poker	☺	😐	☺	☺	😐
SLIM	😐	😐	☺	☺	😐
COCOMO I	☹	😐	☺	☺	😐
COCOMO II	😐	😐	☺	☺	😐
ESTIMACS	☹	😐	☹	☹	☺
PRICE-S	☹	😐	☹	☹	😐
OLS	😐	☺	😐	☺	☺
CART	😐	☺	☹	☺	☺
OSR	😐	☺	☹	☺	☺
ANOVA	😐	☺	😐	😐	☺
COBRA	☺	😐	😐	☺	☺

Table 2 - Estimation Approaches Compared

In order to provide a better understanding of the contents of Table 2, we explain their meaning according to each criterion.

Inputs

☺ - Altran Portugal stores sufficient information to apply this model.

😐 - Altran Portugal has to use the external dataset to cover information gaps in its dataset to use the model.
☹ - Altran Portugal cannot apply this model, without collecting additional information.

Repeatability

☺ - This model produces the same estimates for the same problem, using a well-defined, repeatable process.
😐 - This model has a repeatable, well-defined process, however, it also relies on expert judgment input, which affects the repeatability of the output.
☹ - This model highly depends on expert judgment and does not use a well-defined repeatable process.

Complexity

☺ - This model can be easily used and understood by practitioners.
😐 - This model can be easily used, however, it requires some effort to understand the underlying processes of the model.
☹ - This model requires some statistical expertise to use or keeps the underlying process hidden from the estimator.

Transparency

☺ - The process on which the model relies is publicly available and is well defined.
😐 - The process on which the model relies is publicly available, however it relies on project manager's decisions, rather than a process provided step by step
☹ - The process on which the model relies is not publicly available or it depends on the person performing the estimate.

Accuracy

☺ - This model presents high accuracy in the experimental studies observed.
😐 - This model presents contradictory results in the experimental studies observed or lack of information on its accuracy.
☹ - This model presents low accuracy in the experimental studies observed.

4.3 Candidate prediction approaches identification

To initiate this study, we identified the estimation approaches available in the literature, as discussed in chapter 2. After the identification of the estimation approaches, we provided an overview of each approach, where we also address their limitations and their advantages. For every estimation approach, we studied some of the estimation models that could be applied. We describe how to apply the estimation approaches, the challenges they face, the pros they offer and the characteristics we considered more important to make a comparison between them. Furthermore, we conducted a comparison on the studied estimation approaches, reflected on Table 2. The data in the input column should be considered as a surrogate for the feasibility of applying the specific approach to Altran Portugal's software factory, rather than as a generic classification of the particular approach with respect to its input. In contrast, repeatability, complexity, transparency and accuracy are characteristics inherent to the approaches, rather than to their applicability to the Altran Portugal current context.

Taking into consideration the information presented on Table 2, we can derive some conclusions. With regard to the input information required by each approach, we observed that, the ones based on expert judgment have advantage over the model based approaches, as experts are more flexible, in a way they can receive several types of information to perform the estimation, whereas models expect specific inputs. On the other hand, model based methods present higher repeatability than expert judgment based methods. This can be explained, as it is easier to reproduce automated processes than mental decisions. When addressing complexity, in general, expert judgment approaches are simpler, easier to use and to understand, as the estimation process heavily relies on the opinion of the estimator, rather than in functions and equations. In terms of transparency, property models do not provide the information on which they rely to come up with the estimate, hiding the details from estimators,

constituting a drawback for this type of models. When it comes to accuracy, we stated, through the observation of studies, that specific models, usually outperform general models. This can occur, due to the fact that the accuracy of generic models is strongly affected by the model's calibration to the environment, which can be hard to achieve.

4.4 Gap Analysis to Altran Portugal internal project database

Firstly, it was necessary to acknowledge what information Altran Portugal already stores. This was done through the extraction of the projects Altran Portugal stores to obtain the data they contained. Then, we provide the answer to the first research question.

4.4.1 Assessment to the Information Altran Portugal already possesses

In order to conduct a gap analysis to Altran Portugal project database, we filtered the projects extracted from this database, so that only projects without missing information were taken into account. After applying this filter, we obtained 12 projects. Furthermore, we analyzed each project to extract the information that we can use to feed the estimation models. The projects contain a work breakdown structure, the scheduling of the project, task dependencies and the members in the team. A visual representation of the information a project contains can be seen in Figure 6.

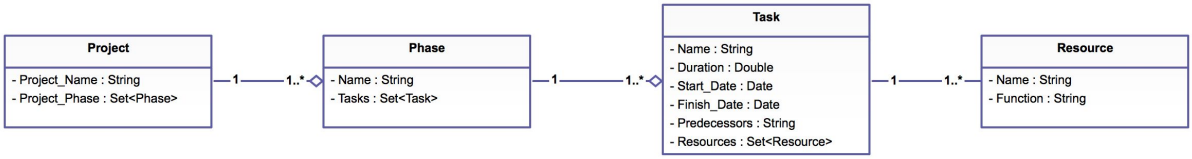

Figure 6 - Project Information

A Project can be characterized by set of phases. Most of Altran projects have 5 project phases in common:

1. Project Management
2. Analysis and Design
3. Development
4. Testing
5. Production

The project management phase consists in the tasks, which aim to prepare and control the execution of the project. Analysis and Design phase take into account user needs, assessing the requirements of the project and developing the design of the system to be developed. The Development phase consists in the development of the software application, using the output of the previous phase as a guide. The testing phase aims to demonstrate that the system conforms to the requirements and the Production phase consists in the deployment of the system into a production environment. Each task has its duration, a start date, a finish date, the tasks that precede it and the resources allocated to it. The information on the resources consists in their names and the role they play in the project. According to this information, the estimation approaches that Altran is currently able to use are:

- Individual Expert Judgment
- Group Review
- Wideband Delphi
- Planning Poker
- CART
- OSR
- OLS
- Stepwise ANOVA
- COBRA

Despite being able to use data-driven models, we expect these to have a weak performance in terms of accuracy, as they highly depend on the dimension of the dataset, and Altran provides a small dataset.

4.4.2 Assessment to the information Altran Portugal is missing

In the case of SLIM and COCOMO I, it would be necessary to store the Delivered Source Instructions in order to estimate the system size.
Regarding PRICE-S and ESTIMACS, the models cannot be applied unless the organization pays for them.
The estimation techniques that Altran Portugal is not able to use are:

- SLIM
- COCOMO I
- COCOMO II
- PRICE-S
- ESTIMACS

The majority of projects that Altran Portugal takes are new, web development projects. The organization uses the waterfall development model and the team size varies from 2 to 10 workers, depending on the size of the project. For future reference, if Altran Portugal stores Delivered Source Instructions and Lines of Code, it will be possible to add SLIM, COCOMO I and COCOMO II to the available estimation techniques.

4.4.3 Answer to RQ1

The gap analysis we conducted lead to the possible estimation approaches:

- Individual Expert Judgment
- Group Review
- Wideband Delphi
- Planning Poker
- COCOMO II
- CART
- OSR
- OLS
- Stepwise ANOVA
- COBRA

In order to provide information about the estimation approaches, so that Altran Portugal can make an informed decision on which to apply, we assess each one individually taking into account Altran's context.
Individual Expert Judgment – This is the approach Altran Portugal uses to perform the estimates. In order to improve the accuracy of the estimates, and to improve the chances of reaching a higher CMMI level, the estimation process must rely more on past project data, rather than in guessing or intuition and define a methodology to record the information of the projects consistently and systematically.
Group Review – This approach shows low repeatability, as each estimator performs the estimate using his own methods. The time and human resources required can also be prohibitive.
Wideband Delphi – Despite having a higher repeatability, this technique can also present high costs in time and human resources.
Planning Poker – This model was created in an agile context, as Altran Portugal uses the waterfall model, it might be an incompatible solution.
COCOMO II – As it is necessary to use the external dataset to make an analogy regarding the system size, it highly depends on the similarity of the projects of the within and cross-company datasets.
CART – To implement this model, the estimators need to know how to make regression trees. Project managers at Altran Portugal might not have such skills.
OSR – This model relies on complex algorithms, which can be a problem for the estimator in the comprehension of model's behavior.
OLS – This model needs a large dataset to be used. Altran possesses a small dataset, which could lead to high estimation errors. In addition, this model is a hard to interpret for non-statisticians.
Stepwise ANOVA – This model is also hard to interpret, which can lead to the resistance of project managers to adopt.

COBRA – This model requires that a group of project managers answer a questionnaire.

5 Altran Database Project Analysis

5.1 Abstract

Altran Portugal, as many other organizations, does not have a tradition of storing information from past projects in a consistent and systematic manner. It does, however, keep an informal project dataset. We used it to bootstrap an internal projects database. As required to achieve the CMMI level 3 goal, this database was then leveraged, in order to increase awareness of how accurate Altran Portugal's estimates are, identify possible effort deviations causes and use historical data to improve estimates.
This analysis is intended to help understanding how the estimating process works, in practice, and to check if the estimates have patterns when grouped by different characteristics.
The projects dataset consisted of 12 projects, which were analyzed individually with respect to their estimate's effort variation, from the first estimate until project completion, in some cases projects were more monitored than others (i.e. the estimate was updated more times), as well as the evolution of the projects' Work Breakdown Structure (WBS), in the same period.
Estimates have higher accuracy in some business areas. A closer look to effort distribution among different project phases shows that, as projects evolve, the effort budget for some phases is typically sacrificed in order to keep the whole project on track, with respect to its total effort. There are programming environments where project managers underestimate the Analysis and Design phase, which is a rare scenario among the other groups. Larger projects tend to have a better control of the global effort by the project manager.
Furthermore confirmation of these observations will require enriching the dataset with other projects from Altran Portugal.
Estimators can use information from past projects according to the size of the project, business area and programming environment in order to use that information as a reference to perform the estimation, bringing more confidence to the estimation.

5.2 Problem Statement

Altran Portugal estimators considers that effort is the most important and challenging variable to estimate, from which cost and schedule estimates are derived. Many organizations often face cost over-runs on their projects, caused by inaccurate initial estimates. These over-runs may lead to smaller profits, or even losses, or imply sacrificing to some extent the quality of the project (e.g. by dropping some non-essential requirements). Significantly increasing the safety margin in estimates is not a viable answer, as it leads to less competitive project bids. Accurate estimates allow presenting better offers when bidding for a project, due to a higher confidence on the estimate, reducing the risk of making a non-sustainable bid. Estimates improvement additionally enables Altran Portugal to deliver software products with higher quality and better budgeting.

5.3 Research Objectives

Overall, we aim to better understand the estimation details in Altran Portugal's projects dataset, and how these evolve, as more project information becomes available. More formally, our goal is to analyze the estimates made for each internal project, for the purpose of finding patterns relevant to reduce estimate deviations, with respect to the effort on each project phase, the project attributes and the WBS changes, from the point of view of a researcher trying to assess the estimates made on the internal projects, in the context of improving the estimates in Altran Portugal.
In particular, this study aims to answer the following research questions:
RQ1: Do project managers transfer effort between phases?
RQ2: Can we find patterns when grouping projects by the programming environment?

RQ3: Can we find patterns when grouping projects by business area?
RQ4: Can we find patterns when grouping projects by size of the project?
RQ5: Do more refined WBSs lead to more accuracy in the estimation?
RQ6: Does a more monitored project lead to less effort deviation?
RQ7: Are there deviations between estimates and the real effort?
For each of the presented research questions, we will formulate a corresponding hypothesis, presented as a pair with the null and its alternative hypothesis.
Hypothesis H1 corresponds to research question RQ1, H2 to RQ2, and so on:
$H1_{null}$: There is no evidence of effort transfer between phases
$H1_{alt}$: There is evidence of effort transfer between phases
$H2_{null}$: We cannot find patterns when aggregating projects by programming environment
$H2_{alt}$: We can find patterns when aggregating projects by programming environment
$H3_{null}$: We cannot find patterns when aggregating projects by business area$H3_{alt}$: We can find patterns when aggregating projects by business area
$H4_{null}$: The size of the project has no relationship with estimate accuracy
$H4_{alt}$: The size of the project is correlated to the accuracy of the estimate
$H5_{null}$: The refinement of the WBS has no relationship with estimation accuracy
$H5_{alt}$: The refinement of the WBS affects estimation accuracy
$H6_{null}$: The constant update of the estimate has no relationship with effort variation
$H6_{alt}$: The constant update of the estimate is correlated to effort variation
$H7_{null}$: There is no evidence of deviations from the estimates to the real effort values
$H7_{alt}$: There is evidence of deviations from the estimates to the real effort values

5.4 Context

Each project manager makes a number of estimates at different times during the project's lifecycle.
A Project can be characterized by a set of phases. All Altran projects have 5 project phases in common:

1. Project Management
2. Analysis and Design
3. Development
4. Testing
5. Production

The project management phase consists in the tasks aimed to prepare and follow-up the execution of the project. Analysis and Design involve eliciting requirements of the project and designing the system to be developed. The Development phase consists in the development of the software application, using the output of the previous phase as input. The testing phase aims to demonstrate that the system conforms to the requirements. Finally, the Production phase consists in the deployment of the system into a production environment and the project follow-up during the first weeks live. Each task has its duration, a start date, a finish date, the preceding tasks, the resources allocated to it and the effort in man-hours.

5.5 Related Studies and Relevance to Practice

According to [20] and [9], having information from previous projects helps the estimator to perform better estimates using the individual expert judgment approach. This analysis is also intended to make information on past projects available, so that estimators can use it to make an analogies, checking how similar a new project is to previous projects, leveraging lessons-learned from previous projects, or even reproducing estimates on some modules or tasks of previous projects in the new estimates.

5.6 Goals

The main goals of this assessment are to understand how accurate the estimates on Altran Portugal projects are, identify estimation patterns, possible deviations and common estimation errors.

5.7 Experimental Units and Material

In order to conduct this analysis, we filtered the projects extracted from Altran Portugal repository so that only projects without missing information were taken into account. After applying this filter, we obtained 12 projects. We also classified these projects using specific attributes, in order to test the hypotheses stated previously:

- Programming Environment
  - . Net, SharePoint, OBIEE, IBM DataStage, Access, Excel VBA, Microsoft BI
- Business Area
  - Human Resources, Knowledge Management, Business Intelligence, Healthcare, Sales Force Automation
- Updated Estimates
  - Updated, Not Updated
- Project Size
  - Small (<1000 man-hours), Medium (1000<man-hours<2000), Large (>2000 man-hours)
- WBS Refinement
  - 2 Levels, 3 Levels, 4 Levels, 5 Levels, 6 Levels

This project contextual information complements the consecutive effort estimates, and actual effort, detailed following the project's WBS. Note that, as a project evolved, it was common to observe that its WBS evolved, as well.
In order to test hypothesis 1, it is necessary to assess if there is a decrease on one phase and an increase on other phase in terms of effort, comparing the estimate with the real values.
Hypothesis 2 is tested using the Programming Environment grouping, comparing the effort variation of each group of projects.
In order to test hypothesis 3, we use the Business Area grouping, comparing the effort variation of each group of projects.
To test Hypothesis 4, we use the Project Size grouping, comparing the effort variation of each group of projects.
To test hypothesis 5, we use the WBS Refinement grouping, comparing the effort variation of the 5 levels of refinement.
Hypothesis 6 is tested using the Updated Estimates grouping, assessing if estimates that were updated presented less effort variation between the first and the final estimate when compared to estimates that were not updated and to assess if the estimators are decreasing the difference between the effort from the first estimate to the real effort value.
Hypothesis 7 is tested observing the several estimation versions in order to compare the effort values to assess if there are deviations from the first version of the estimate to the final version, representing the real effort values.

5.8 Procedure and Procedure Analysis

Altran Portugal internal project files were exported from Microsoft Project to Microsoft Excel format and from Excel to IBM SPSS Statistics, the tool used for data analysis.
In order to understand how the estimation process evolved, we analyzed the effort's variation and the modifications made to the WBS for every estimate performed on each project. Furthermore, we compared the estimated effort with the real effort according to the projects grouped by a specific attribute, observing the effort at each phase to identify relevant patterns using boxplots. A boxplot is a tool used in data analysis that offers an efficient way to examine a dataset in order to have a visual representation of the distribution of that data. The box consists in a vertical rectangle that is divided by

a line denoting the median that separates the first quartile from the third quartile. The first quartile is the middle number between the median and the smallest value on the dataset and the third quartile is the middle number between the median and the highest value in the dataset. Two lines extended from the top and bottom of the box called whiskers denote the highest and the lowest non-outlier value observed respectively. Finally, the asterisks represent the outlier values found on the dataset.
We used the Kruskal-Wallis, the Mann-Whitney and Wilcoxon statistical tests, as we are going to compare distributions that are not normal. The normality tests for each test are shown on the Append section.

5.9 Execution

Altran Portugal facilitated data collection by providing all the projects already filtered (i.e. they excluded the projects for which there was relevant data missing). However, the data had to go through several format transformations before being imported to the SPSS tool, which revealed to be a very time-consuming task.
We studied each project individually, according to the changes made to the WBS and the effort deviation. This first assessment is intended to detect unusual, unexpected or interesting events that occurred in the projects. Then, we analyze a set containing all the projects, which enable us to understand what is currently happening in terms of effort on each lifecycle phase. We proceed to isolate projects according to their Programming Environment, Business Area, Updated Estimates, Project Size and WBS Refinement and analyze them according to the effort spent on each phase of their project lifecycle concerning their effort estimated values and their real effort values. Finally, we test the hypotheses and present the results.

Project	Programming environment	Business Area	Tasks Added	Tasks Removed	Tasks Split	Tasks Merged	Effort Variation (Man-hours)	Estimate Updates
1	Custom Development .Net	HR	9 15.5%	0	3 5.2%	0	+53 +4.6%	1
2	Custom Development .NET	HR	26 26.0%	9 9.0%	0	2 2.0%	+865.76 +20.9%	0
3	SharePoint	Knowledge Management	0	0	0	0	0	0
4	OBIEE	BI	19 13.8%	15 10.9%	1 0.7%	0	+844 +54.9%	4
5	IBM DataStage	BI	19 16.7%	13 11.4%	0	19 16.7%	+1177.9 +91.9%	0
6	OBIEE	BI	60 31.1%	11 5.7%	1 0.5%	2 1.0%	+2753.2 +85.6%	0
7	Custom Development Access	Healthcare	0	0	0	0	+88 +6.2%	0
8	Custom Development .NET	Healthcare	140 134.6%	74 71.2%	8 7.7%	1 1.0%	+2236.8 +66.1%	0
9	Excel VBA	SFA	7 31.8%	4 18.2%	0	0	+37.2 +7.8%	2
10	Custom Development .NET	BI	8 29.6%	2 7.4%	0	0	+290.53 +59.6%	4
11	Microsoft BI	BI	0	0	0	0	-48 -13.8%	0
12	OBIEE	BI	19 11.0%	10 5.8%	0	0	+1445.6 +52.0%	6

Table 3 - Project Data

5.10 Analysis

The assessment on the Altran Portugal's internal Projects will enable us to answer the research questions outlined in section 5.3.
Table 3 shows the programming environment on which the project relied, the business area regarding the project, the modifications to the WBS in terms of tasks and the variation of the effort between the first and final estimate. There are seven types of programming environments represented on the projects, five business areas and only five projects had their estimates updated. The number of estimate updates performed varies from 0 to 6.

Project 3 has its initial and its final estimate equal and the tasks and respective effort is the same, this might indicate that the project manager did not save the estimation data, saving only the real effort values and final WBS.
Another estimate that caught our attention was the one referent to project 5. This project had its development phase highly underestimated, as we can see in Figure 7, being V1.0 the estimated effort value and V1.1 the real effort value. Despite the fact that posterior phases suffered a reduction on their effort, the project still presented a big effort variation. The effort percentage estimated to be spent on the development phase was of 40% but the real effort percentage spent turned out to be around 120%, as seen in Table 3. This means that the real effort spent on the development phase is higher than the total effort of the first estimate on the project.

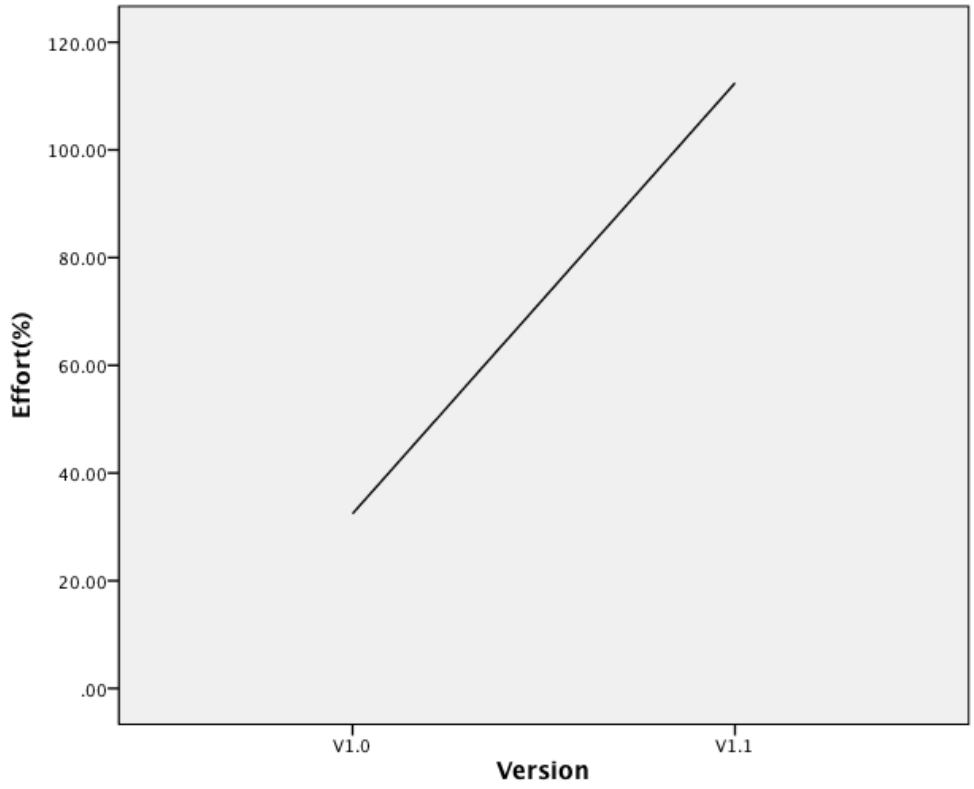

Figure 7 - Development Effort Variation for Project 5

Project 8 suffered major changes in its work breakdown structure. A great number of tasks was added (140) and deleted (74) throughout the project, which contributed to the variation of the estimated effort, which was 66.1% more than expected.
Project 9 is the only project whose effort variation presented a negative value (i.e. the real effort for concluding the project was lower than what had been estimated) and the real effort was the closest to the predicted effort. This project had 4 estimate versions and the Development phase was divided in two parts (Development-I and Development-II). The first version corresponds to the initial estimate and the last version to the real effort spent on the project. As we can see in Figure 8, the Development-I phase had a negative value from the first to the second version, which indicates that the perception of effort needed for this phase lowered. However, we can see that from the second to the third version, the same phase suffered an increase in the effort expected to fulfill its tasks, turning from negative values to an increase around +20%. The project management phase was clearly underestimated in the first version, presenting an increase of 20%. Finally, the phase Development-II turned out to present an increase of about 50% on the real values (final version), which indicates that the estimator only realized the deviations on this phase after project completion.
An interesting fact is the sequence of the modifications on the versions of the project. It seems that these modifications were made according to the time order that each phase takes on the project. The Project Management phase was the first to suffer significant changes, then the Development-I and

finally the Development-II. Development-I changed at the same time as Project Management, however, it needed to be recalculated later, which supports the sequence of phases through time on a project.

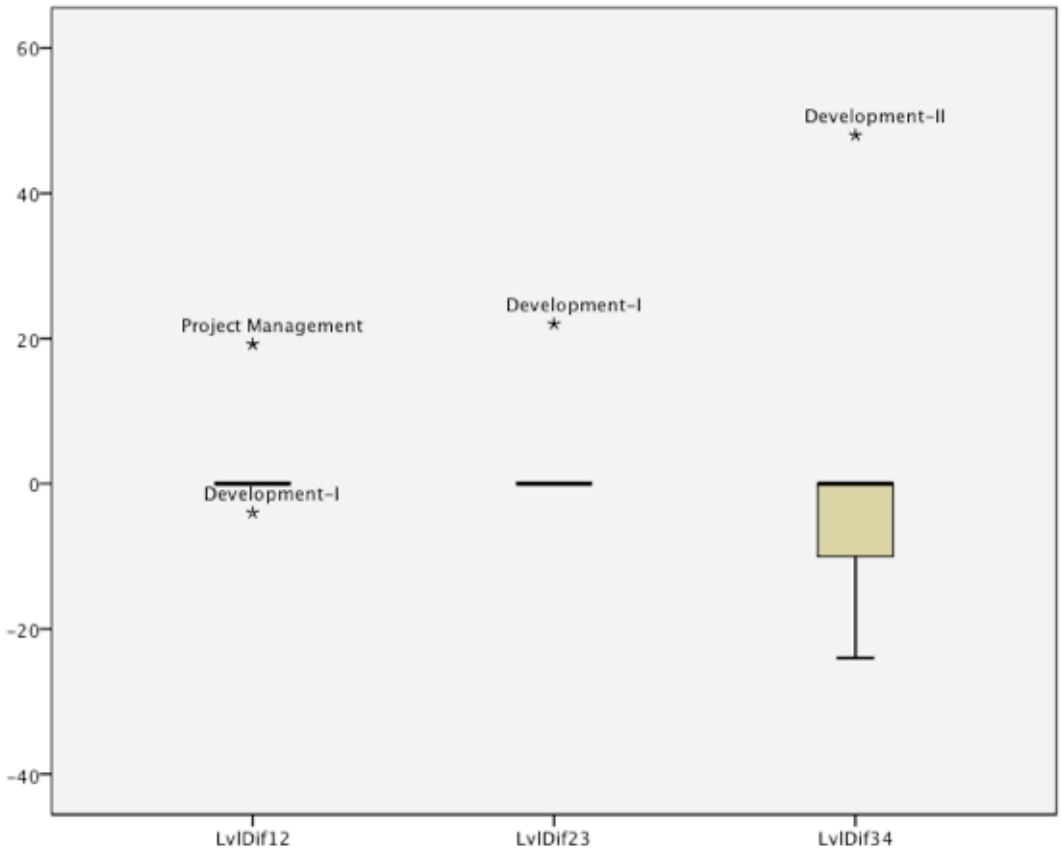

Figure 8 - Project 9 estimate versions comparison according to the effort %

After analyzing the projects that presented the most distinctive characteristics of particular projects, we assess how the projects were estimated at first, on their effort, and compare the values with the real effort later observed. Figure 9 on the left shows the effort breakdown in terms of percentage of all the initial effort estimates for each main phase of the projects whereas on the right shows similar data, showing the real effort values instead of the estimated.

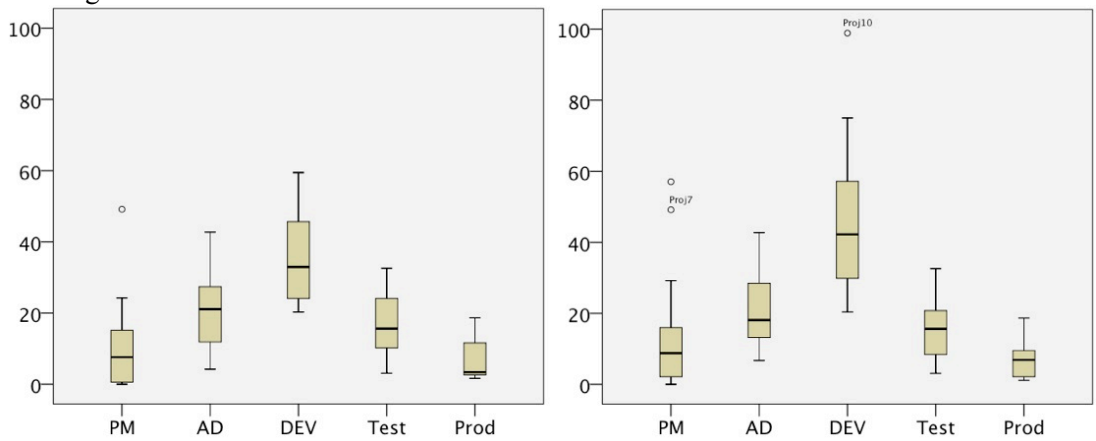

Figure 9 – Effort % by phase (Estimate)

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	7[a]	5.29	37.00
	Positive Ranks	2[b]	4.00	8.00
	Ties	3[c]		
	Total	12		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-1.718[b]
Asymp. Sig. (2-tailed)	.086

a. Wilcoxon Signed Ranks Test
b. Based on positive ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Estimated_Effort and Real_Effort equals 0.	Related-Samples Wilcoxon Signed Rank Test	.086	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 10 - Wilcoxon test (Estimated, Real) for Development

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	6[a]	6.50	39.00
	Positive Ranks	4[b]	4.00	16.00
	Ties	2[c]		
	Total	12		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-1.173[b]
Asymp. Sig. (2-tailed)	.241

a. Wilcoxon Signed Ranks Test
b. Based on positive ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Estimated_Effort and Real_Effort equals 0.	Related-Samples Wilcoxon Signed Rank Test	.241	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 11 - Wilcoxon test (Estimated, Real) for Analysis and Design

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	4[a]	4.50	18.00
	Positive Ranks	2[b]	1.50	3.00
	Ties	6[c]		
	Total	12		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-1.572[b]
Asymp. Sig. (2-tailed)	.116

a. Wilcoxon Signed Ranks Test
b. Based on positive ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Estimated_Effort and Real_Effort equals 0.	Related-Samples Wilcoxon Signed Rank Test	.116	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 12 - Wilcoxon (Estimated, Real) for Production

The effort on the analysis and design phase estimated was higher than the real effort spent. Conversely, the effort estimated on the Development and Production phases was lower than the real effort spent.

However, as we can see in Figure 11 there were more projects underestimated (6) than overestimated (4) on the analysis and design phase. Figure 10 and Figure 12 also show that the development and production phases had more projects underestimated than overestimated.

5.10.1 H1 – Effort transfer between phases

The effort that an estimator allocates to a lifecycle phase can later be adjusted to compensate some deviations that may occur and balance the total effort of the project. To transfer effort between phases it is necessary to remove tasks from a phase and add tasks to another phase, which happens in 8 projects according to Table 3. However, the tasks might be added and removed from the same phase. To understand how, in general, the project's effort evolved on each phase, we assessed Figure 9 to state that the Analysis and Design phase suffers a decrease in the effort percentage and the Development and Production phases increase.
We want to test the hypothesis that the effort is overestimated in earlier lifecycle phases (Analysis and Design) to balance the underestimated effort on later ones (Development and Production). In other words, we are testing the hypothesis that no transferences of effort occur.
Effort was transferred from the analysis and design phase to other phases. However, the statistical tests on the development phase (Figure 10), on the analysis and design phase (Figure 11) and on the production phase (Figure 12) indicate that the median of differences between the estimated and the real effort on each phase is zero. This means that the differences of effort are not statistically significant and automatically leads us to accept the null hypothesis that there are no transfers of effort between phases and reject $H1_{alt}$.

5.10.2 H2 - Programming Environment

Concerning the Programming Environment, we have found that on .Net projects, the Analysis and Design phase effort percentage increased from the initial estimate to the real value and the development phase shows a very high variance on the third quartile as we can see in Figure 13. We tried to find patterns on this type of projects because it is one of the two categories that possesses more than 1 project and the underestimation of the Analysis and design phase is underestimated, which represents half of the times it occurred in the dataset, representing a possible pattern.
Other Programming environments do not present significant patterns or consist in only one representative (Figure 14). We want to test the hypothesis that, aggregating projects according to the Programming Environment does not show estimation patterns.
The Wilcoxon test in Figure 15 shows that the differences of effort on the analysis and design phase of .Net projects are not statistically significant and lead us to accept the null hypothesis that there are no patterns regarding this programming environment and reject $H2_{alt}$.

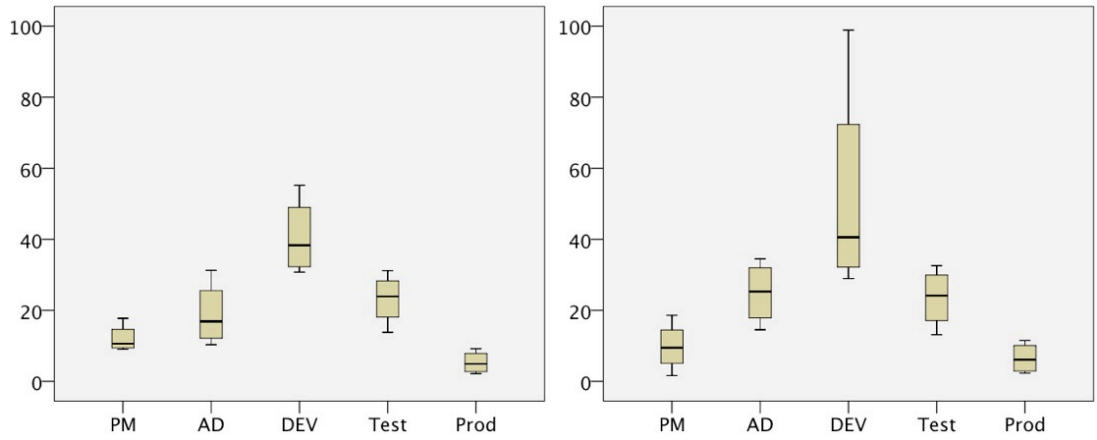

Figure 13 - Effort by phase % (left: Estimate, right: real) .Net projects

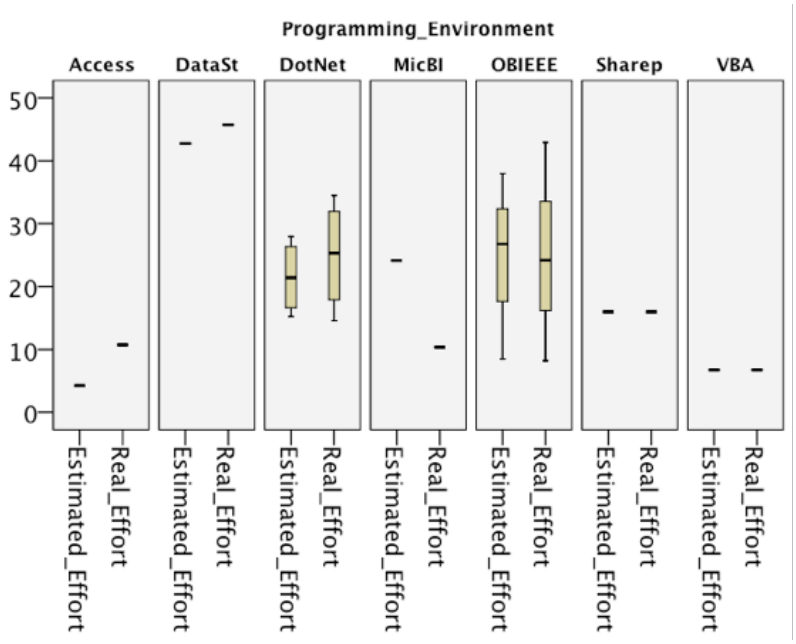

Figure 14 - Effort % for Analysis and Design by Programming Environment

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	3[a]	2.67	8.00
	Positive Ranks	1[b]	2.00	2.00
	Ties	0[c]		
	Total	4		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-1.095[b]
Asymp. Sig. (2-tailed)	.273

a. Wilcoxon Signed Ranks Test
b. Based on positive ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Estimated_Effort and Real_Effort equals 0.	Related-Samples Wilcoxon Signed Rank Test	.273	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 15 - Wilcoxon (Estimated, Real) for analysis and Design on .Net projects

5.10.3 H3 - Business Area

The results on the Business Area show that on Business Intelligence projects, the Analysis and Design phase shows more variance (Figure 16) and the Development phase costs more than the estimated and presents a very high variance, in terms of effort, whereas on Healthcare projects this phase presented the opposite behavior as shown in Figure 18, which is a rare occurrence. Healthcare projects also show that the Production phase needs more effort than it is estimated, this can be seen in Figure 17.

We want to test the hypothesis that, aggregating projects according to the business area does not show estimation patterns regarding this type of projects.

The Wilcoxon test made on healthcare projects shows that the difference on the development phase of the estimated and real effort is not statistically significant (Figure 20). On the other hand, the test we made on business intelligence projects shows that there is a difference on the development phase between the medians of the real and estimated effort (Figure 19), which lead us to reject the null hypothesis and accept $H3_{alt}$. In other words, there are estimation patterns when aggregating projects by business area, as BI projects typically underestimate the development phase.

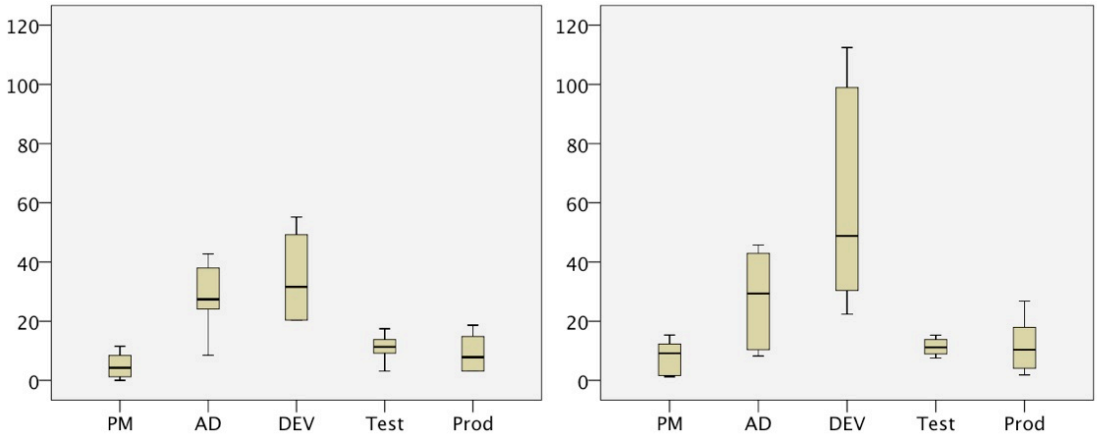

Figure 16 - Effort % by phase (left: Estimate, right: Real) BI

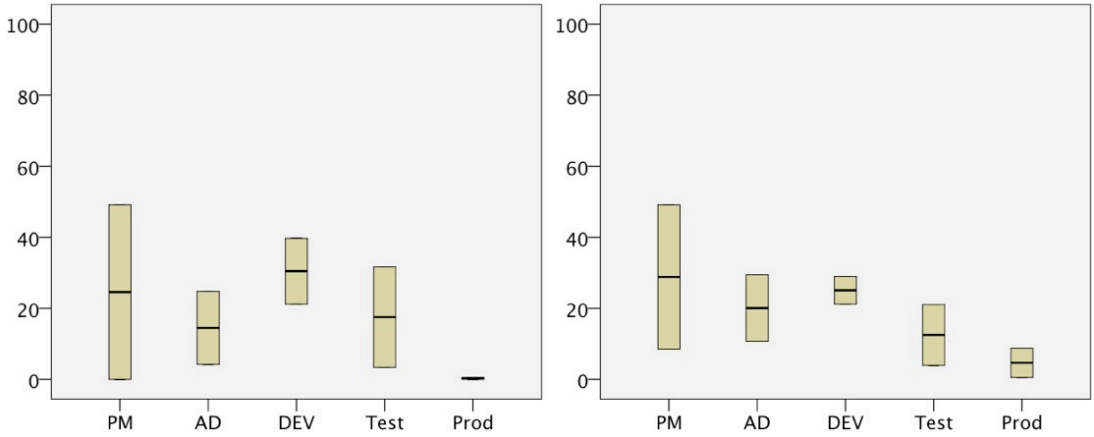

Figure 17 - Effort % by phase (left: Estimate, right: Real) Healthcare

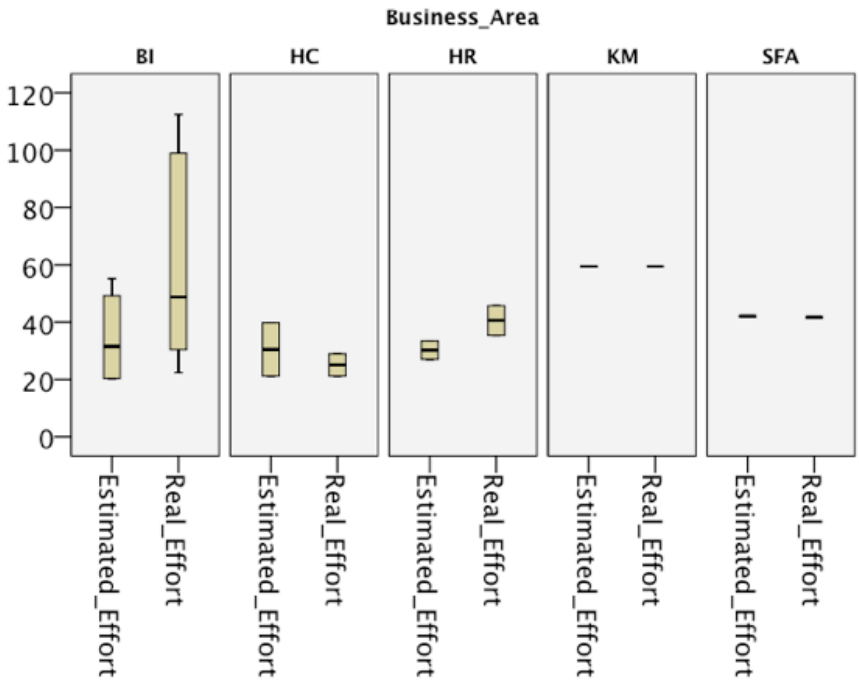

Figure 18 - Effort % for development by Business Area

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	5[a]	3.00	15.00
	Positive Ranks	0[b]	.00	.00
	Ties	1[c]		
	Total	6		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-2.023[b]
Asymp. Sig. (2-tailed)	.043

a. Wilcoxon Signed Ranks Test
b. Based on positive ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Estimated_Effort and Real_Effort equals 0.	Related-Samples Wilcoxon Signed Rank Test	.043	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 19 - Wilcoxon (Estimated, Real) for development on BI projects

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	0[a]	.00	.00
	Positive Ranks	1[b]	1.00	1.00
	Ties	1[c]		
	Total	2		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-1.000[b]
Asymp. Sig. (2-tailed)	.317

a. Wilcoxon Signed Ranks Test
b. Based on negative ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Estimated_Effort and Real_Effort equals 0.	Related-Samples Wilcoxon Signed Rank Test	.317	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 20 - Wilcoxon (Estimated, Real) for Development on Healthcare

5.10.4 H4 - Project Size

On small size projects (Figure 21), the Analysis and Design phase costs less effort than it was estimated, on the other hand, the development phase presented higher percentage of effort on the real values in contrast with the estimated values, the real values also presented higher variation on this phase.

On medium size projects (Figure 22), the development phase presented higher percentage of effort on the real values in contrast with the estimated values, the real values on this phase also presented higher variation of effort, that variation is inferior when compared to the one observed on small projects, however, medium size projects present outliers.

On large size projects (Figure 23), the Analysis and Design phase costs more effort than it was estimated, it is also stated that the variation of the effort is inferior when compared with medium size projects.

We want to test the hypothesis that, aggregating projects according to their size is not significant to study them in terms of effort accuracy.

These Kruskal-Wallis test made to the size of the project shows that the error (Real effort – Estimated effort) is not statistically significant across the size categories, leading us to accept the null hypothesis and reject $H4_{alt}$. This means that the size of the project does not influence the accuracy of the estimate.

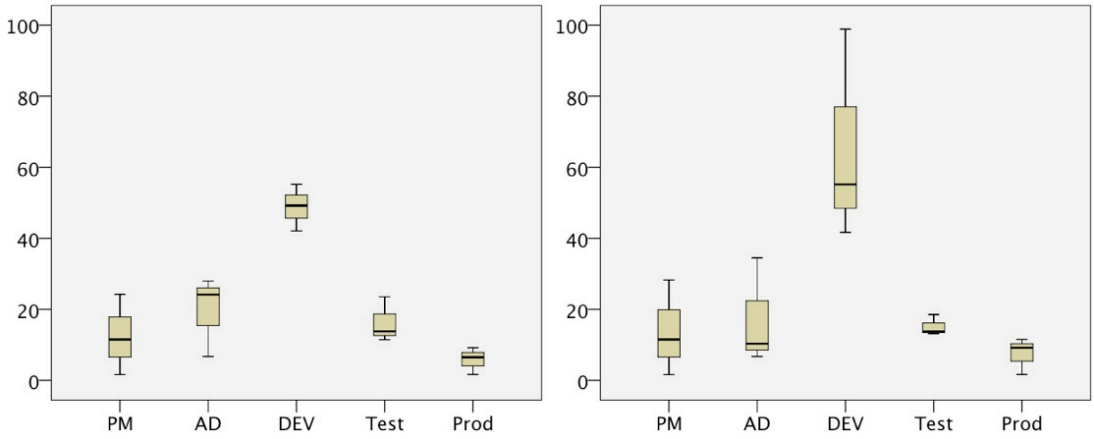

Figure 21 - Effort % by phase (left: Estimate, right: Real) Small

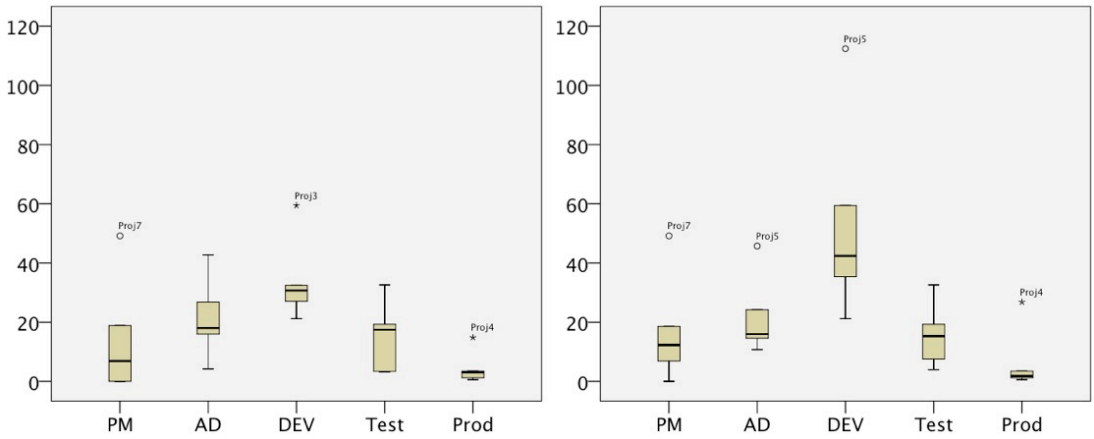

Figure 22 - Effort % by phase (left: Estimate, right: Real) Medium

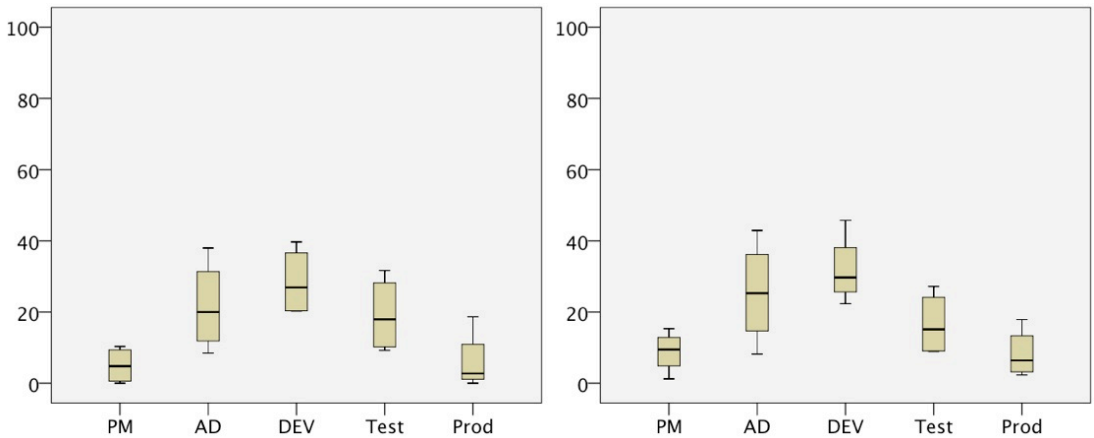

Figure 23 - Effort % by phase (left: Estimate, right: Real) Large

Kruskal-Wallis Test

Ranks

	Size	N	Mean Rank
Error	1	3	6.00
	2	5	5.20
	3	4	8.50
	Total	12	

Test Statistics[a,b]

	Error
Chi-Square	1.938
df	2
Asymp. Sig.	.379

a. Kruskal Wallis Test

b. Grouping Variable: Size

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Error is the same across categories of Size.	Independent-Samples Kruskal-Wallis Test	.379	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 24 - Kruskal-Wallis (Error) on Size (1-Small, 2-Med, 3-Large)

5.10.5 H5 - WBS Refinement

Estimates with 3 levels on the WBS (Figure 25) presented some outliers, like project 7 which had a project management which was very high in terms of effort and had its testing phase overestimated, clearly indicating the drop of some tasks as it is around 0%. It is also interesting that project 10 had its production phase underestimated, and the effort needed to finish this phase was much higher than the rest on the 3 level category.

The Development phase on the estimate represented less than 40% of effort values. The real value was around 60%.

When the WBS has 4 levels (Figure 26), we observed that the analysis and design phase presented effort percentages around 25% on the estimate. As for the real value, this phase had an effort lower than 20% with higher variance. The development and testing phase presented higher effort on the real values. However it also presented a lower variance.

In 5 level WBS projects (Figure 27), the development phase involved a lower effort on the real values when compared to the estimates.

If the WBS presents 6 levels (Figure 28), we state that the effort estimated on the Analysis and Design phase is lower than the effort spent in reality, as well as on the Development phase.

Despite showing less variance, Level 6 WBS estimates are a lot similar to the Level 5 ones, being similar in terms of effort deviation.

We want to test the hypothesis that, the refinement of the WBS does not lead to less effort variation.

The Kruskal-Wallis test in Figure 29 shows that the distribution of error (Real effort – Estimated effort) is the same across WBS levels, which lead us to accept the null hypothesis that the refinement of the WBS does not lead to less error and reject $H5_{alt}$.

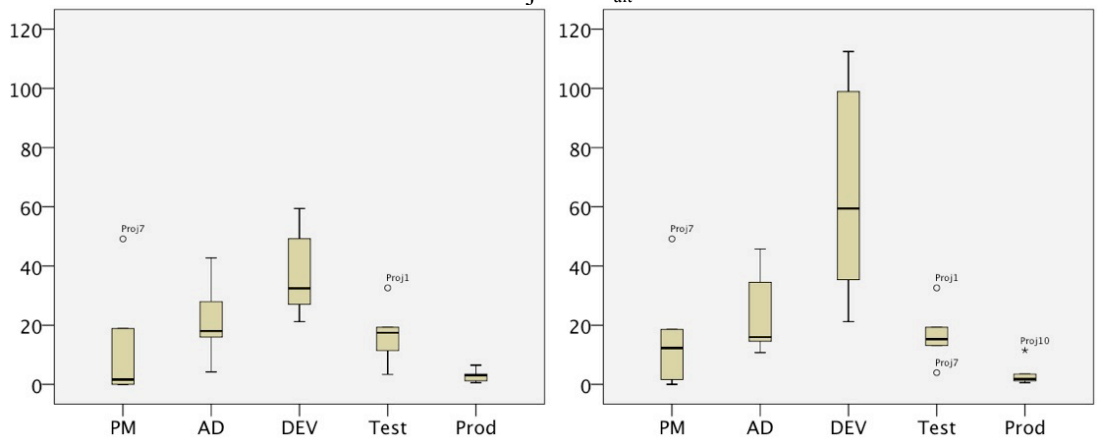

Figure 25 - Effort % by phase (left: Estimate, right: Real) 3 Levels

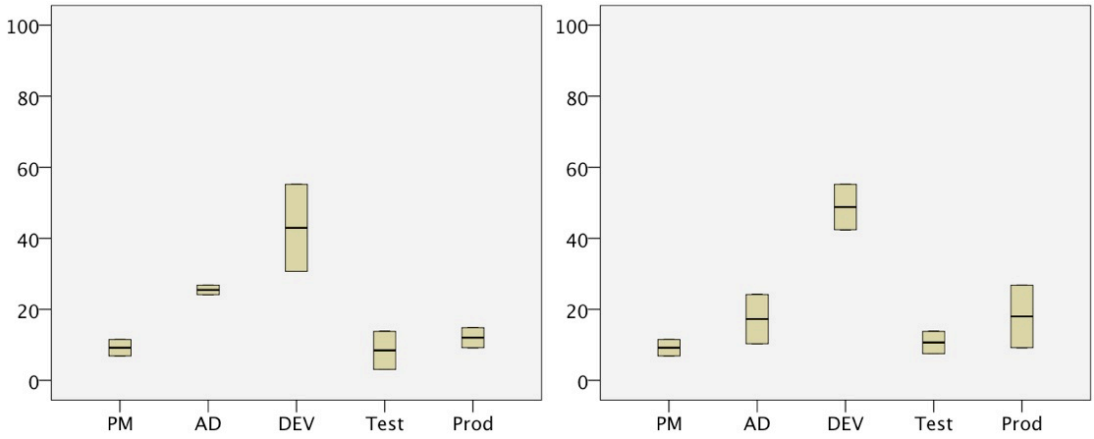

Figure 26 - Effort % by phase (left: Estimate, right: Real) 4 Levels

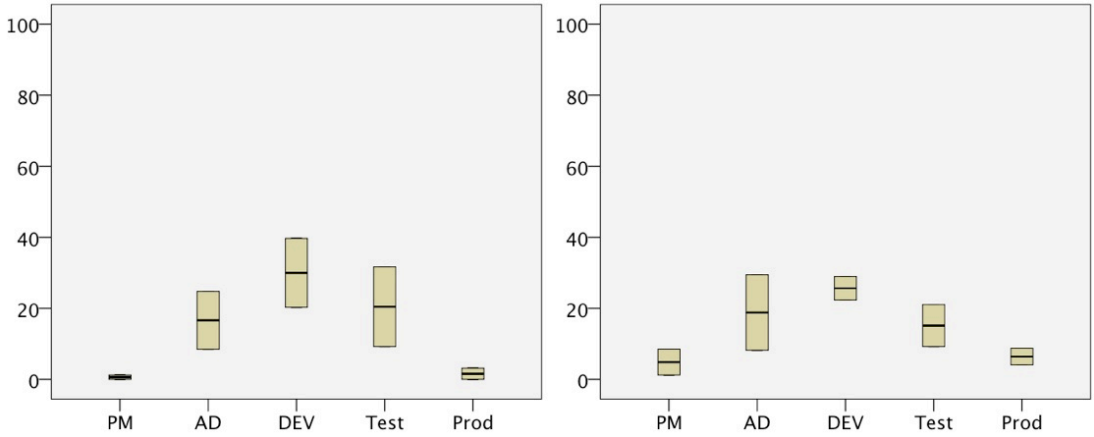

Figure 27 - Effort % by phase (left: Estimate, right: Real) 5 Levels

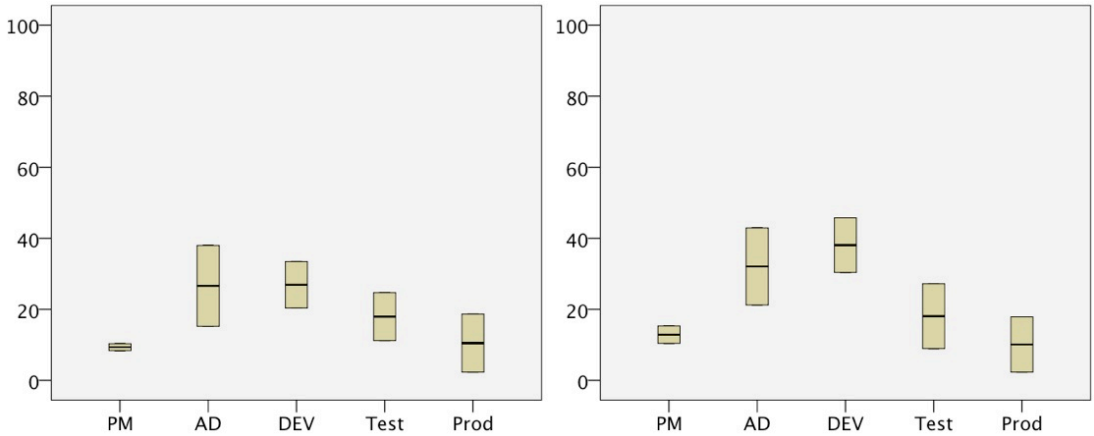

Figure 28 - Effort % by phase (left: Estimate, right: Real) 6 Levels

Kruskal-Wallis Test

Ranks

	WBS	N	Mean Rank
Error	2	1	4.00
	3	5	5.40
	4	2	6.50
	5	2	8.50
	6	2	8.50
	Total	12	

Test Statistics[a,b]

	Error
Chi-Square	2.177
df	4
Asymp. Sig.	.703

a. Kruskal Wallis Test

b. Grouping Variable: WBS

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Error is the same across categories of WBS.	Independent-Samples Kruskal-Wallis Test	.703	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 29 - Kruskal-Wallis (Error) on WBS level

5.10.6 H6 - Estimate Update

By updating the estimate, estimators may make adjustments to the project, like adding and removing tasks to make compensations in terms of effort, balancing the total effort of the project and minor the deviations.

It was observed that on estimates which were not updated (Figure 30), the percentage of effort estimated for each task does not present a significant difference to the real effort spent with the exception for an outlier on the development phase.

On projects where the estimator updated the estimates (Figure 31), there was not a significant difference between the estimated and real version of the estimate with the exception of the outliers on the development and production phase. On the other hand, we can state that projects that had their estimates updated presented a convergence in the direction of the real effort values as seen in Figure 32 and Figure 33.

We want to understand if the frequent update of the estimate does not lead to the convergence of the real effort values. As the project progresses, we expect the estimate updates to narrow the gap between the predicted and the actual values.

To answer RQ6, we observed that the update of the estimates leads to a convergence on the real values, even if between two versions the estimated effort deviates from the real effort, the last estimate is always closer to the real value than the first estimate.

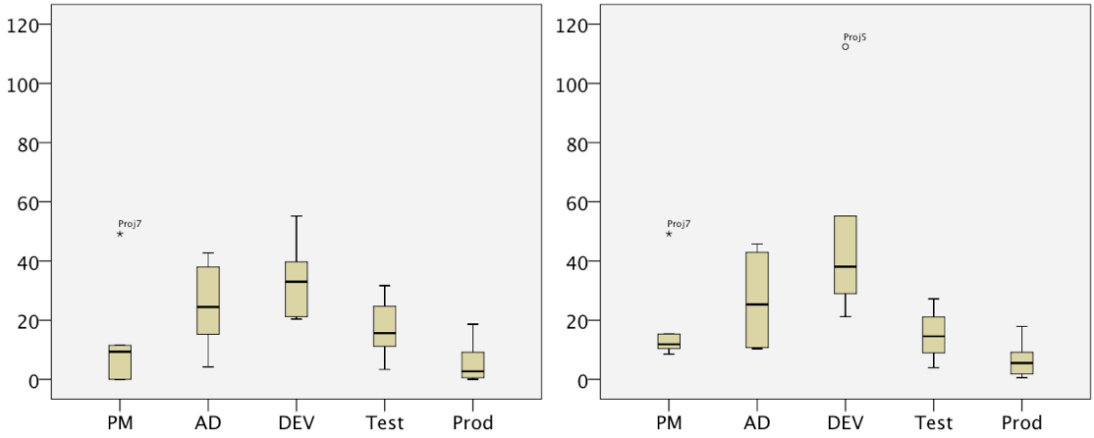

Figure 30 - Effort % by phase (left: Estimate, right: Real) Not Up

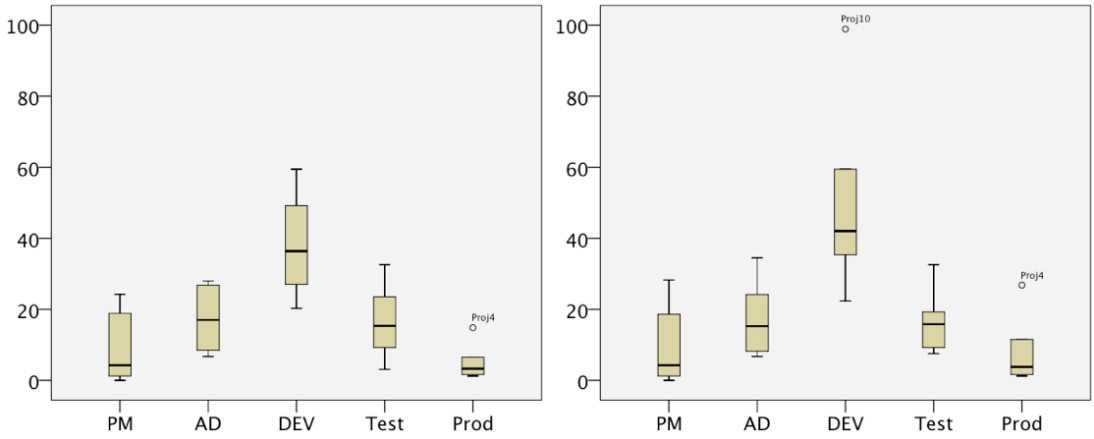

Figure 31 - Effort % by phase (left: Estimate, right: Real) Updated

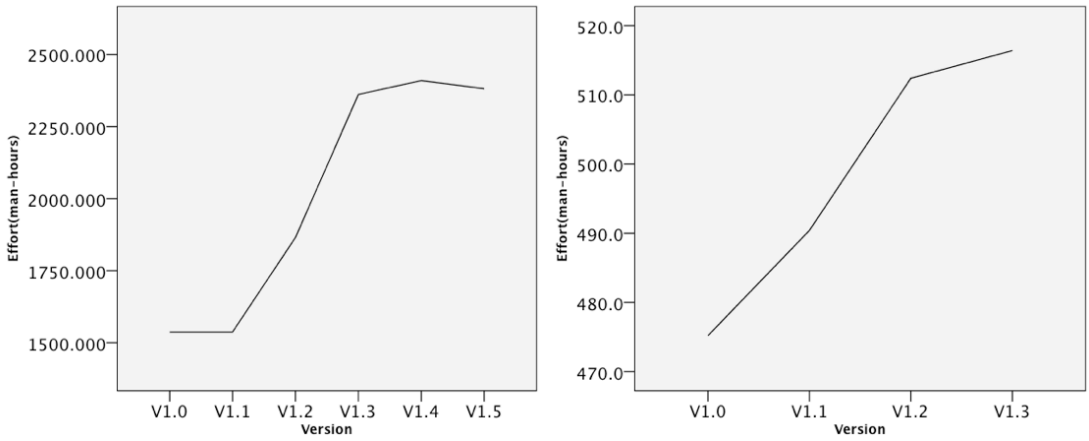

Figure 32 - Estimate versions (left: Project 4, right: Project 9)

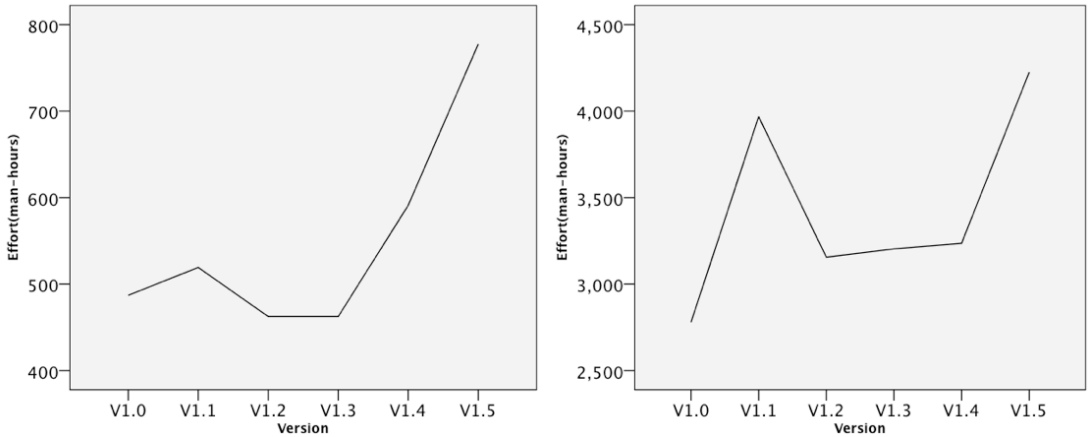

Figure 33 – Estimate versions (left: Project 10, right: Project 12)

5.10.7 H7 – Deviations

The observation and comparison of the first and last version of the estimate of the projects represented in Figure 32 and Figure 33 enable us to acknowledge that the effort estimated in the first version of the estimate is not equal to the real effort that is spent on the project.

We want to test the hypothesis that there are no differences between the estimated effort and the real value of each project.

Table 3 shows that every project, with exception of project 3 had their real effort values different from the estimated, ranging from -13.8% to 91.9%, which clearly means that there are deviations. However, Project 3 is the exception, as we stated earlier, this may happen due to being just the record of the real project effort values instead of the estimate versions as well.

The Wilcoxon test in Figure 34 leads us to reject the null hypothesis that there are no effort differences between the estimated values and the real values and reject $H7_{alt}$.

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	10[a]	6.40	64.00
	Positive Ranks	1[b]	2.00	2.00
	Ties	1[c]		
	Total	12		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-2.756[b]
Asymp. Sig. (2-tailed)	.006

a. Wilcoxon Signed Ranks Test
b. Based on positive ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Estimated_Effort and Real_Effort equals 0.	Related-Samples Wilcoxon Signed Rank Test	.006	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 34 - Wilcoxon (Estimated, Real) on all projects

5.11 Inferences

5.11.1 Overall Projects

When we compare Figure 9, we observe that the initial estimate tends to be optimistic in terms of effort. However the phase AD (Analysis and Design) shows the opposite, which more probably indicates that estimators transfer effort from this phase to the remaining phases, in order to minimize the impact on the total effort that the project requires. We can also state that the critical phase is the development (DEV), as it represents the most percentage of effort spent on the project. We also observe a major difference between the estimated and the real effort values. This phase is the most challenging when performing the estimation, as it is most of the times underestimated. In addition, the production phase (Prod) is also underestimated, as its mean of effort percentage on the project increases. However, the testing phase (Test) does not change significantly. This may happen due to good estimation but it is also possible that the company may prefer to deliver the product faster and more error prone, in this case, some tasks on the testing phase might have been dropped off to reduce the effort of the project. Sometimes it is necessary for companies to sacrifice some of the quality of the product in order to deliver it before the deadline or to meet the budget.

5.11.2 Individual Project analysis

Project 3 shows no changes on the WBS and also no changes when comparing the estimate with the real vales. This may indicate that this is not really an estimate, but just the recording of the project data.

5.11.3 Programming Environment

In .Net projects, the analysis and design phase is underestimated, which is not common, as it usually decreases in order to compensate the necessary increase on other phases. This hints that this is a particularly important phase on this type of projects.

5.11.4 Business Area

In BI projects, the effort might be hard to estimate, as the Development phase is highly underestimated. The opposite situation happens in Healthcare projects, which is very unusual, in the sense that the Development phase is rarely overestimated. This would make it a good candidate for further scrutiny, to determine why overestimation is so less of a problem in projects for this particular domain.

5.11.5 Size

It is visible that the estimates are more accurate proportionally to the size of the project. This might happen because the importance of the project can be connected to its size, leading to higher attention and care on the behalf of the project manager. The time that the project takes is also higher, which also allows the project manager to take mitigation actions, when deviations are detected.

5.11.6 WBS Refinement

As for the WBS Refinement, our assessment showed that the refinement of the WBS leads to a higher accuracy on the estimate. We are constrained to the number of breakdown levels used in the projects, but relatively similar performance of estimations with 5 and 6 levels suggests it may not be necessary to consider further decomposition levels. On the other hand, it was also possible to state that the despite increasing it is not significant in a statistical manner.

5.11.7 Estimate Update

The update of the estimation does not seem to be a relevant factor when searching for estimation patterns, however, estimators that update the estimates will decrease the effort difference between the estimated and real value as perception of the project increases.

5.12 Threats To Validity

The fact that the dataset is composed by a small number of projects (12) has a negative impact on a statistical relevance level, as we observed when testing the hypotheses.
The fact that the internal projects did not have more than 6 levels does not let us assess if the estimates could present higher accuracy when even more refined.

5.13 Answer to RQ2

We can state that estimators generally underestimate the effort required to finish the project. It also seems that estimators use (in some cases) the analysis and design phase to balance the effort with posterior lifecycle phases, such as the development and production phase. However, we could not prove this phenomenon, as there was no statistical data to prove it.
Business intelligence projects tend to overestimate the effort spent in the development phase.
We tried to prove that the size of the project would influence the accuracy of the estimate of the effort with no success, however the observation of the boxplots in section 5.10.4 lead us to suspect that it happened.
We also tried to prove that the WBS refinement leads to less effort variation, which also could not be statistically proved despite the patterns we found on the boxplots in section 5.10.5.
It was also possible to state that estimators that update a previous estimate reduce the distance to the real effort. It does not always happen a convergence from version to version, however, the final estimate is always closer to the real effort than the first one.
Finally, it is important to emphasize that the volume of data on this database was not big enough to eliminate our pattern suspicions or to statistically prove them. However, since it was not possible to prove some of the hypothesis tested, it is also relevant to say that some of the patterns we were testing were not strong enough to turn it into statistical evidence.

6 Altran Evolution projects analysis

6.1 Abstract

Altran Portugal has software evolution projects, characterized by a large number of evolution requests over a relatively long period. It is important to improve the estimates in order to be able to decrease the effort deviations and it is crucial to provide high quality service. As for the projects described in the previous chapter, it is also necessary to systematically make estimates concerning these change requests. Increasing the accuracy of these estimates is a stepping-stone toward improving the control over these projects. In the long run, Altran Portugal aims to decrease the technical debt – the time spent satisfying requests after the deadline – to decrease the response time taken to address those requests and to raise the awareness on the actual time spent fulfilling those requests.
We assessed the differences between the estimated and real effort according to the priority, category and complexity of the change request and also to the presence of Java in the resolution of the change request (as this is perceived in the company as a source of additional complexity). Moreover, we calculated the response time, technical debt and effective time bound to each project, and assessed their correlation to the priority of the project, the complexity and the existence of a Java component on the project. Finally, we used the data provided to study the viability of using a forecasting model.
We observed that the estimators tend to underestimate the effort required to fulfill change requests. We also observed that the effort increases with the complexity of the request and that, indeed, requests with a Java component typically require more effort than the ones without it. We also stated that the technical debt decreases for higher priority projects, just as the response time. The effective time spent increases when a project has a Java component and, as one would expect, also for more complex projects.
Although there is a relatively large number of change requests in the analyzed project, the time frame of this project is, so far, too small so that time series analysis can be effectively used to make forecasts. A longer period of data collection is required for enabling this kind of analysis.

6.2 Problem Statement

Altran Portugal strives for constant improvement. It is necessary to present this conduct not only to beat the competition but also to achieve client satisfaction. The client whose evolution projects we are assessing has recently renewed the trust it had on Altran Portugal and has made another solicitation to have its requests for evolution handled by the organization for an extended period of time. A systematic improvement increases the confidence clients have in Altran Portugal and enables the company to understand the key factors that influence the effort, leading to a higher awareness of those factors and an increased accuracy of the estimates.
This will contribute to the decrease of the technical debt, which is the time the client is waiting for an evolution request satisfaction when it has passed the deadline, the awareness of the effective time that an evolution request needs to be fulfilled, and the notion of the response time that the team is able to support when a new evolution request arrives.

6.3 Research Objectives

This study aims to answer the following research questions:
RQ1: Are the estimators underestimating or overestimating the effort involved in satisfying evolution requests?
RQ2: Does the category of the project influence the effort needed?
RQ3: Is the effort of the project related to its complexity?
RQ4: Do projects with a Java component require more effort?

RQ5: Does the technical debt decrease for higher priority projects?
RQ6: Does the response time decrease for higher priority projects?
RQ7: Does the effective time depend on the complexity of the project?
RQ8: Does the effective time increase when the project has a Java component?
RQ9: Is the dataset fit to use a forecasting model?
Our goal is to analyze the estimates made for evolution projects, for the purpose of understanding what are the key factors that influence the effort, with respect to the project characteristics and measures, from the point of view of a researcher trying to assess the estimates made on the evolution projects that a client requested, in the context of improving the estimates in Altran Portugal.
For each of the presented research questions, we will formulate a corresponding hypothesis, presented as a pair with the null and its alternative hypothesis.
$H1_{null}$: The estimators are producing correct effort estimations
$H1_{alt}$: The estimators are not producing correct effort estimations
$H2_{null}$: There is no evidence that the effort is affected by the category
$H2_{alt}$: There is evidence that the effort is affected by the category
$H3_{null}$: There is no evidence that the effort is affected by the complexity
$H3_{alt}$: There is evidence that the effort is affected by the complexity
$H4_{null}$: There is no evidence that projects with a Java component require more effort
$H4_{alt}$: There is evidence that projects with a Java component require more effort
$H5_{null}$: There is no evidence that the technical debt decreases for higher priority projects
$H5_{alt}$: There is evidence that the technical debt decreases for higher priority projects
$H6_{null}$: There is no evidence that the response time decreases for higher priority projects
$H6_{alt}$: There is evidence that the response time decreases for higher priority projects
$H7_{null}$: There is no evidence that the effective time is affected by the complexity of the project
$H7_{alt}$: There is evidence that the effective time is affected by the complexity of the project
$H8_{null}$: There is no evidence that the effective time increases for projects with a Java component
$H8_{alt}$: There is evidence that the effective time increases for projects with a Java component
$H9_{null}$: The dataset is not fit to use a forecasting model
$H9_{alt}$: The dataset is fit to use a forecasting model

6.4 Context

When the client wants to make a request for evolution, a unique identifier is given to the evolution request, along with a summary, the priority, the category and the complexity of the project. As for the time, the project has information on the date when the project was transferred, the date of the initial response, the deadline and the date of project closure. In terms of effort, the projects provide the estimated effort and the real effort in hours.
With this information we can calculate the technical debt (closure date – delivery date), the effective time that is needed to finish a project (closure date – initial response date) and the response time (Initial response date – date when project was transferred).

6.5 Related Studies and Relevance to Practice

Boehm [9] presents the cone of uncertainty (already discussed in Chapter 1, and presented in Figure 1), where, as the understanding of the project grows, the level of uncertainty decreases. This is originally targeted to development projects, rather than for evolution projects. This distinction is important because, unlike development projects, evolution projects have a moving target, with respect to completion. So, we are not necessarily closing towards the end of those projects when satisfying a change request. Nevertheless, this study enables us to assess if estimates are becoming more accurate due to the increased accumulated information to become more accurate, and will also enable the estimator to have access to more information on the project related to key factors that have impact on the total effort spent on satisfying the evolution requests and improve the estimation accuracy.

6.6 Goals

The main goals of this study are to understand the key factors that are relevant to the estimation of the effort, to assess how the team is behaving considering the technical debt, the effective time needed to finish the projects, the response time given and also to understand the evolution of estimation accuracy.

6.7 Experimental Units and Material

This assessment was made through the study of a dataset containing 219 evolution requests from an Altran Portugal's evolution project. In order to test the hypotheses stated previously, we used the following independent variables, taken from the change requests repository (a more detailed description is presented in Table 4, section 6.4):

- Complexity
  - Low, Medium, High
- Category
  - Internal Applications, Web applications, Exclusive use TSP
- Java
  - True, False
- Priority
  - Normal, Urgent

We also used these dependent variables:

- Estimated Effort
  - Man-hours
- Real Effort
  - Man-hours

Finally, we calculated the following dependent variables:

- Technical debt
  - Hours
- Effective time
  - Hours
- Response time
  - Hours

A visual representation of a request for evolution can be seen in Figure 35.

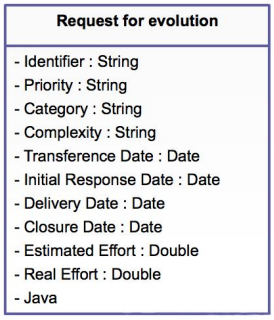

Figure 35 - Request for evolution data

To test hypothesis 1, we compare the estimated effort values with the real effort values and check if the estimated values are significantly different from the real values.
Hypothesis 2 is tested through the comparison of the effort values observed for each category.
Hypothesis 3 is tested through the comparison of the effort values observed for each complexity degree.
Hypothesis 4 is tested observing the effort values spent on requests that involve working with a Java component and compare them with projects without this kind of component.

To test hypothesis 5, we use the calculated technical debt and compare the technical debt values among the different priority categories.
Hypothesis 6 is tested using the response time and comparing its values among different priority categories.
To test hypothesis 7, we use the calculated effective time and compare the effective time values among the different complexity degrees of the projects.
Hypothesis 8 is tested through the comparison of the actual effective time values on projects that have a Java component and the ones that do not have.
Hypothesis 9 is tested creating a model that uses effort values aggregated according to a period of time and observing the values of the effort through time.
To test hypothesis 10, we use the effort values aggregated according to a period of time and assess if the difference between the estimated and real values is decreasing.

6.8 Procedure and Procedure Analysis

The evolution projects provided by Altran Portugal's client were exported from the Microsoft Excel format to IBM SPSS Statistics, which we used to analyze the data.
In order to assess if the estimators are overestimating or underestimating, we compared the estimated effort values with the real effort values. We also compared these values grouping the dataset using the attributes: complexity, priority, category and java. Then, we calculated the technical debt, effective time and response time using the information existent in the dataset. Furthermore, we compared the technical debt and response time between the types of priority and also compared the effective time between the types of complexity and the presence of a Java component.
Finally, we used a time series to model the effort distribution over time, to assess the viability of producing a forecast for its evolution using time series analysis.
Similarly to the internal projects, we also used the Kruskal-Wallis, Mann-Whitney, and Wilcoxon statistical test, in order to compare distributions that are not normal. The normality tests for each test are shown on the Append section.

6.9 Execution

These projects were extracted from a database and transformed to a format suitable for being imported to the SPSS tool.
We began studying the differences between the estimated effort and the real effort values. The next step was to analyze the effort isolating the requests using the attributes priority, category, complexity and Java. After this, the technical debt, response time and effective time were calculated. The technical debt was calculated through the difference between the request closure date and the delivery date. The response time was calculated through the difference between the initial response date made and the date the request was transferred. The effective time can be obtained by the difference between the closure data and the initial response date.
Finally, we test the hypotheses and analyzed the results obtained.

6.10 Analysis

The assessment on the evolution projects dataset enables us to answer the research questions reported in 6.3.
Table 4 shows the evolution requests attributes that were used to conduct this analysis. The identifier is a unique code given to each request, the priority is used to denote if a request is urgent or not, the category represents the type of request, the complexity denotes how complex is the request, the transference date is the date the request was received, the initial response date is the date the request began, the delivery date is the deadline to deliver the request, the closure date is the date when the request was delivered, the estimated effort is the effort estimated to finish the request, the real effort is the effective effort that the project request cost, the java attribute is used to distinguish the requests

that had a java component from those that did not, the technical debt is the time between the deadline and the delivery, the effective time is the effective time spent on the request and the response time is the time between the reception of the request and the its initiation.
Each request has a unique identifier. In terms of priority it can be normal or urgent. There are 3 categories of requests, the internal applications, web applications and exclusive use TPS (transaction processing systems), the complexity of a request can be low, normal or high, the transference date and the closure date include the year, month, day and time (h: m), the estimated effort and the real effort are measured in hours, the java attribute is affected with true if the project contains a java component and false otherwise, the technical debt, effective time and response time are measured in hours.

Identifier	CO(Change order) plus 5 numbers
Priority	Normal, Urgent
Category	Internal Applications, Web Applications, Exclusive Use TPS
Complexity	Low, Medium, High
Transference date	Date (day-month-year hour:minute:second)
Initial response date	Date (day-month-year hour:minute:second)
Delivery date	Date (day-month-year hour:minute:second)
Closure date	Date (day-month-year hour:minute:second)
Estimated effort	Time (man-hours)
Real effort	Time (man-hours)
Java	True, False
Technical debt	Time (hours:minutes)
Effective time	Time (hours:minutes)
Response time	Time (hours:minutes)

Table 4 - Evolution requests data

6.10.1 H1 – Underestimating or overestimating

We assess if the estimators are more conservative, overestimating the effort needed to finish the request or if they are more prone to underestimate the effort needed to fulfill the request. As we can see in Figure 36, the real effort values are superior to the estimated effort values, indicating that the requests are generally underestimated in terms of effort.
We want to test the hypothesis that the estimators are underestimating the effort, predicting more man-hours than the needed to fulfill the request.
We did a Wilcoxon test (Figure 37), showing that 100% of the projects were underestimated, which lead us to reject the null hypothesis that the estimators are producing correct estimations and accept $H1_{alt}$.

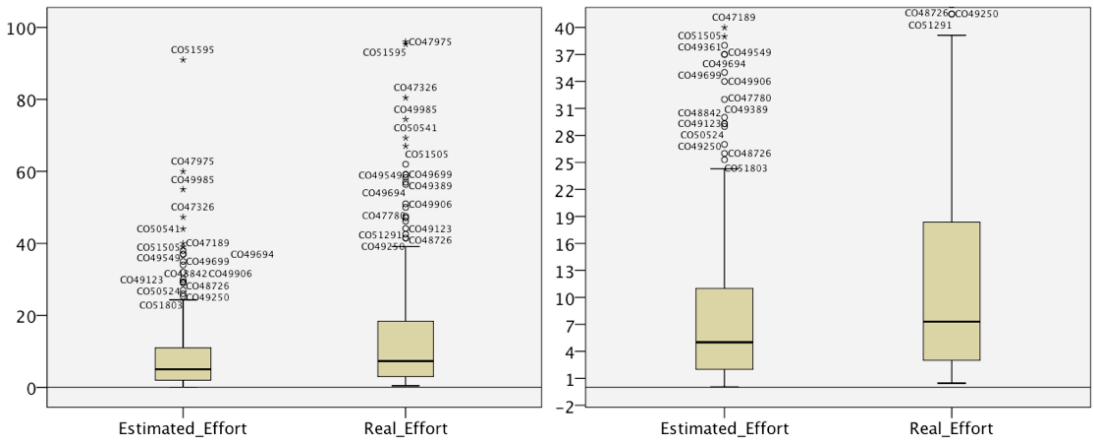

Figure 36 - Estimated and real effort (man-hours) and close-up

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
Estimated_Effort – Real_Effort	Negative Ranks	219[a]	110.00	24090.00
	Positive Ranks	0[b]	.00	.00
	Ties	0[c]		
	Total	219		

a. Estimated_Effort < Real_Effort
b. Estimated_Effort > Real_Effort
c. Estimated_Effort = Real_Effort

Test Statistics[a]

	Estimated_Effort – Real_Effort
Z	-12.831[b]
Asymp. Sig. (2-tailed)	.000

a. Wilcoxon Signed Ranks Test
b. Based on positive ranks.

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The median of differences between Total_Effort and Estimate equals 0.	Related-Samples Wilcoxon Signed Rank Test	.000	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 37 - Wilcoxon test (Estimated effort, Real effort)

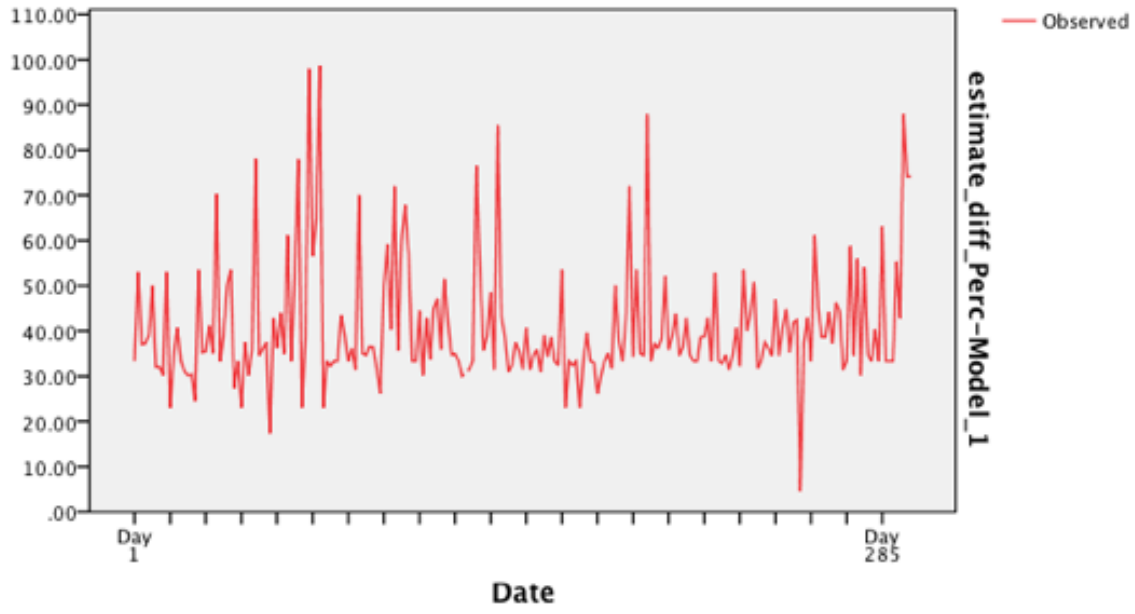

Figure 38 – Difference (%) between estimated and real effort through time

6.10.2 H2 – Category

When we group requests by category (Figure 39), we can state that despite needing less effort to finish the projects, the *internal applications* category shows several outliers. On the other hand, *exclusive use tps* and *web applications* categories show similar effort values and fewer outliers.

We can also observe in Figure 39 that the categories *exclusive use tps* and *web applications* show similar effort patterns, however, the internal applications category demands less effort and presents much more outliers. This indicates that the categories *exclusive tps* and *web applications* can be merged to make comparisons with the internal applications category as they present similar behavior in terms of effort.

We want to test if the effort values are affected by the category of the request, assessing if the effort spent on each category is different or not.

We did a Kruskal-Wallis test (Figure 40) to check if the distribution of effort is the same across all categories, which lead us to reject the null hypothesis that there is no evidence that the category affects the effort spent and accept $H2_{alt}$.

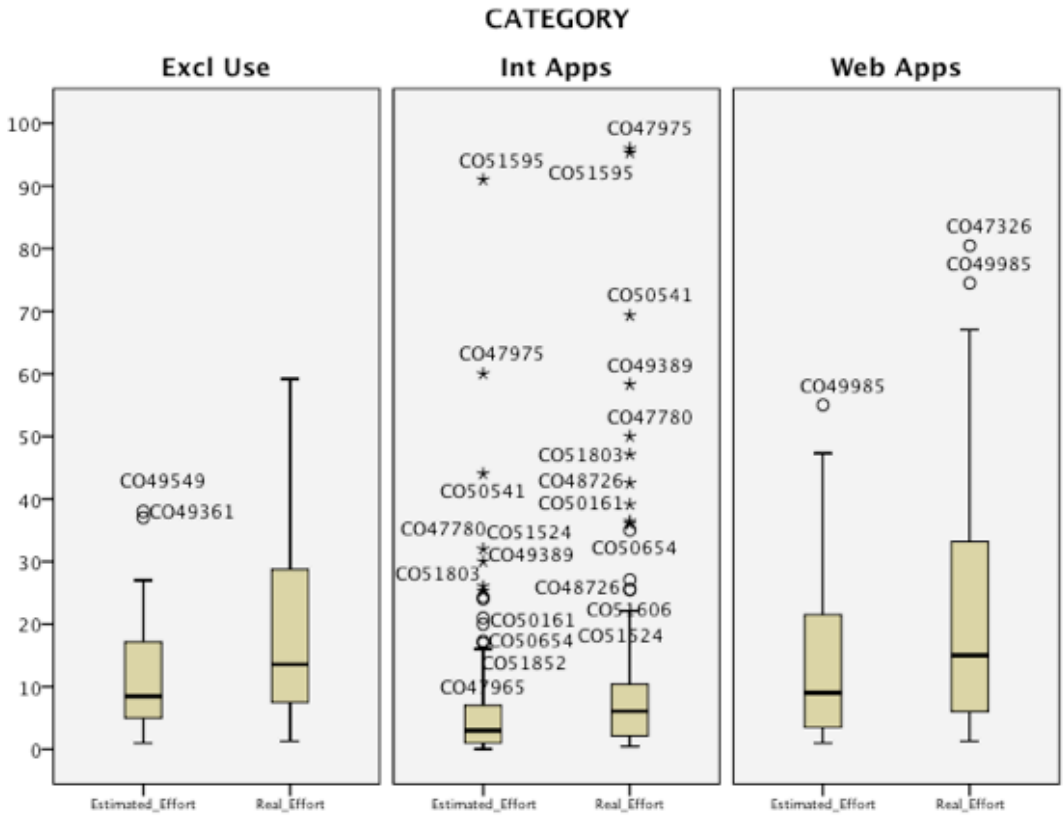

Figure 39 - Estimate and real effort on each category

Kruskal-Wallis Test

Ranks

	COD_CAT	N	Mean Rank
Real_Effort	1	55	138.72
	2	139	92.79
	3	25	142.50
	Total	219	
Estimated_Effort	1	55	139.51
	2	139	92.20
	3	25	144.06
	Total	219	

Test Statistics[a,b]

	Real_Effort	Estimated_Effort
Chi-Square	28.133	30.283
df	2	2
Asymp. Sig.	.000	.000

a. Kruskal Wallis Test

b. Grouping Variable: COD_CAT

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Total_Effort is the same across categories of COD_CAT.	Independent-Samples Kruskal-Wallis Test	.000	Reject the null hypothesis.
2	The distribution of Estimate is the same across categories of COD_CAT.	Independent-Samples Kruskal-Wallis Test	.000	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 40 - Kruskal-Wallis (Estimated, Real) on category (1-Web, 2-Excl Use, 3-Int)

6.10.3 H3 – Complexity

When grouping requests by their assigned complexity level to assess the effort values, we obtain Figure 41. According to this figure, we can observe that low complexity requests require less effort than the medium complexity requests and these require less effort than the high complexity requests. This means that the higher the complexity, the more effort is demanded by the project, as expected.
It is interesting to observe that low complexity requests have more outliers, which may indicate that the some requests might have been wrongly labeled in terms of complexity. It is also interesting to see

that there is a high complexity outlier that had its effort estimated on between 5 and 10 man-hours. This suggests that a request is not necessarily less complex when it requires as much as many man-hours as a low complexity request (in other words, although complexity is often used as a surrogate for the associated effort, this is not always observed, in practice, requests perceived as very complex can actually be solved with a relatively low effort).

We want to test if the effort is affected by the complexity of the request, assessing if the effort spent on each complexity level is different or not.

The Kruskal-Wallis test (Figure 42) we executed shows that the distribution of effort is different among the complexity levels, which lead us to reject the null hypothesis and accept $H3_{alt}$.

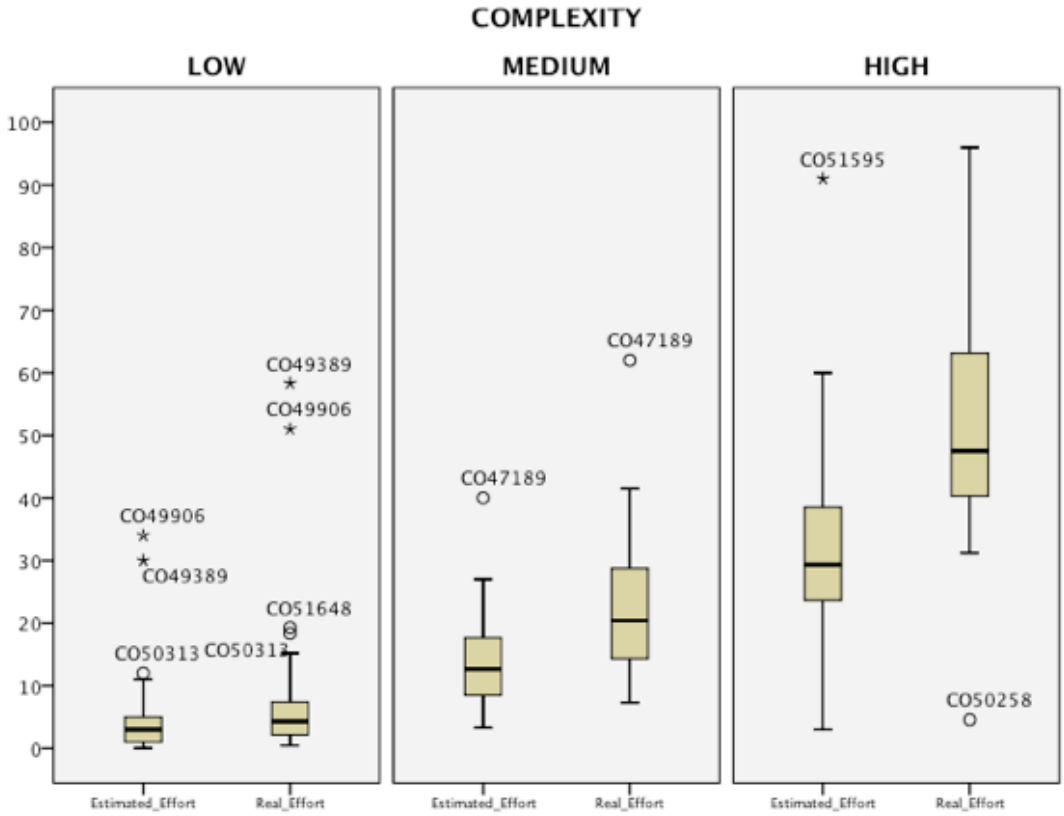

Figure 41 - Estimate and real effort values on each complexity level

Kruskal-Wallis Test

Ranks

	Complexity_CODE	N	Mean Rank
Real_Effort	1.00	147	76.83
	2.00	48	166.40
	3.00	23	199.57
	Total	218	
Estimated_Effort	1.00	147	77.18
	2.00	48	165.61
	3.00	24	199.79
	Total	219	

Test Statistics[a,b]

	Real_Effort	Estimated_Effort
Chi-Square	125.413	125.267
df	2	2
Asymp. Sig.	.000	.000

a. Kruskal Wallis Test

b. Grouping Variable: Complexity_CODE

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Total_Effort is the same across categories of Complexity_CODE.	Independent-Samples Kruskal-Wallis Test	.000	Reject the null hypothesis.
2	The distribution of Estimate is the same across categories of Complexity_CODE.	Independent-Samples Kruskal-Wallis Test	.000	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 42 - Kruskal-Wallis (Estimated, Real) on complexity (1-Low, 2-Med, 3-High)

6.10.4 H4 – Java

Figure 43 shows that when a request has a Java component the effort values are higher. Conversely, the number of outliers is lower than on requests without a Java component.
We want to test if the effort of requests that have a Java component is higher than the requests without it.
We did a Mann-Whitney test (Figure 44) to check if the distribution of effort is the same across the presence and absence of java that lead us to reject the null hypothesis that there is no evidence that requests with a Java component require more effort than the other requests and accept $H4_{alt}$.

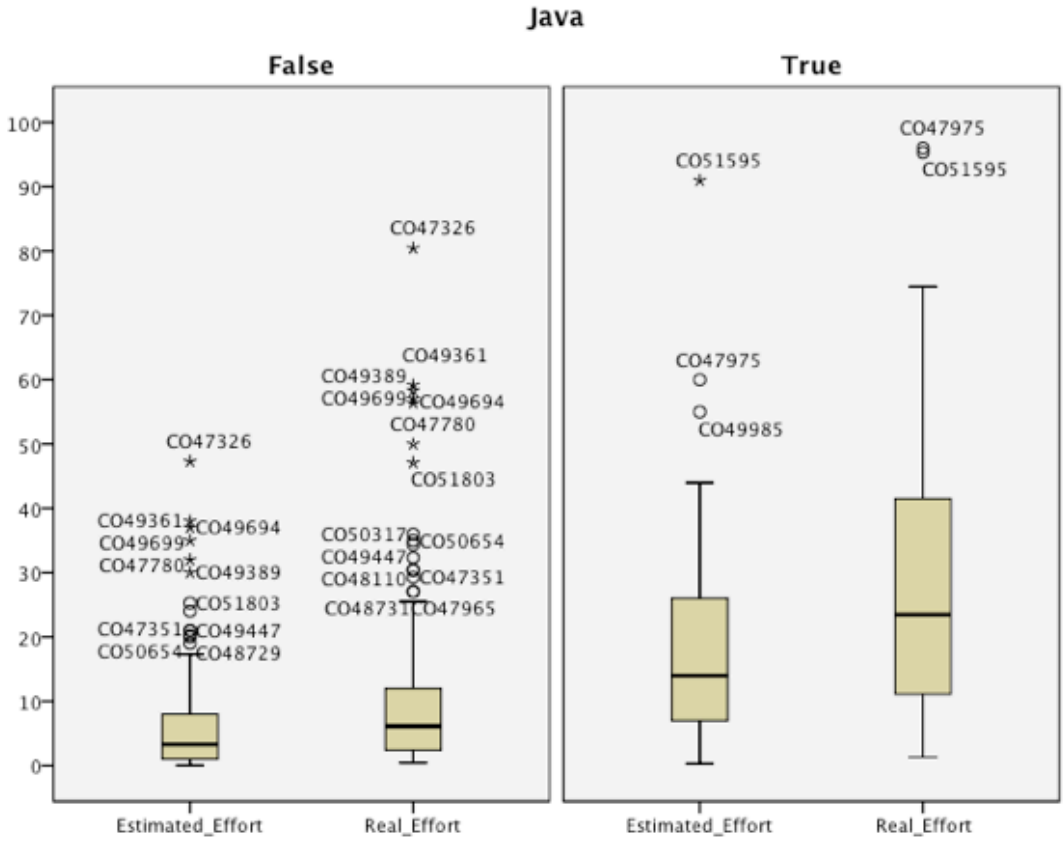

Figure 43 - Estimate and real effort on requests with (true) and without (false) java

Mann-Whitney Test

Ranks

	Java_CODE	N	Mean Rank	Sum of Ranks
Real_Effort	.00	169	95.28	16101.50
	1.00	49	158.56	7769.50
	Total	218		
Estimated_Effort	.00	170	95.69	16267.50
	1.00	49	159.64	7822.50
	Total	219		

Test Statistics[a]

	Real_Effort	Estimated_Effort
Mann-Whitney U	1736.500	1732.500
Wilcoxon W	16101.500	16267.500
Z	-6.185	-6.241
Asymp. Sig. (2-tailed)	.000	.000

a. Grouping Variable: Java_CODE

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Total_Effort is the same across categories of Java_CODE.	Independent-Samples Mann-Whitney U Test	.000	Reject the null hypothesis.
2	The distribution of Estimate is the same across categories of Java_CODE.	Independent-Samples Mann-Whitney U Test	.000	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 44 - Mann-Whitney (Estimated, Real) on Java (0-without Java, 1-with java)

6.10.5 H5 –Technical debt

In general, the technical debt is around 300 hours as we can see in Figure 45. It is also relevant to note that the most distant outlier is almost 10 times greater than the upper quartile limit in terms of hours.

When grouping the projects by priority, we can observe that the technical debt is lower on urgent projects (Figure 46), although, still, around 200 hours.

The closer the technical debt is to 0, the closer it is from the deadline, which means that higher priority requests show less technical debt.

We want to test if higher priority requests suffer less technical debt, in other words, we want to test if requests with higher priority are delivered closer to the deadline than lower priority requests.

We did a Mann-Whitney test (Figure 47) to assess if the distribution of technical debt is the same depending on the priority on the project, which lead us to reject the null hypothesis that there is no evidence that the technical debt decreases for higher priority requests and accept $H5_{alt}$.

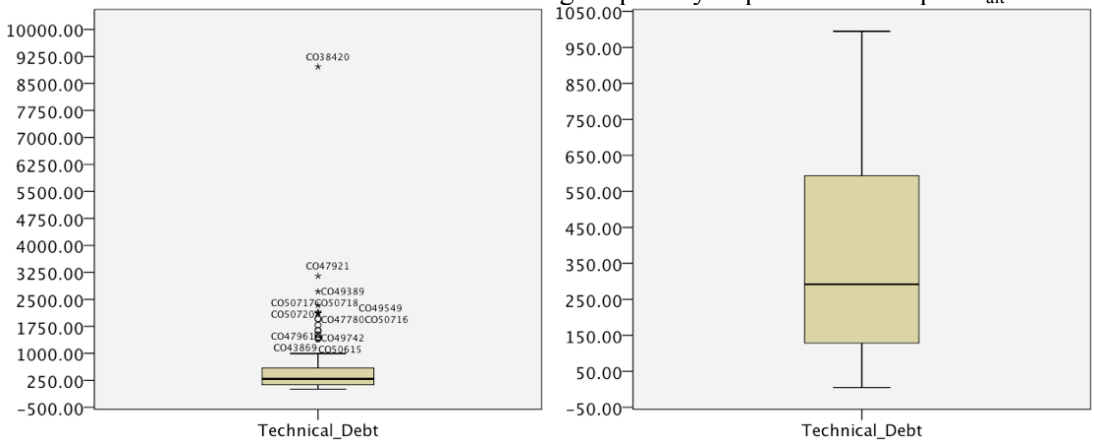

Figure 45 - Technical debt (hours) and close-up

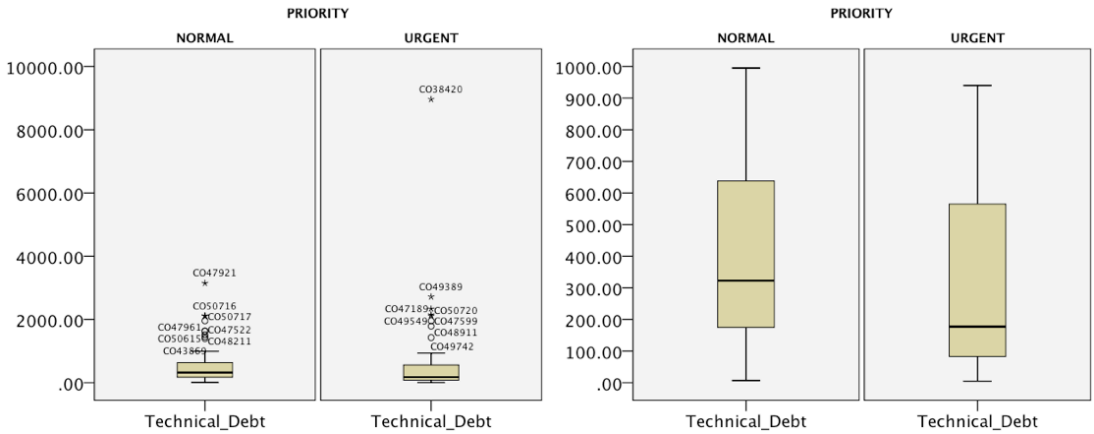

Figure 46 - Technical debt by priority (hours) and close-up

Mann-Whitney Test

Ranks

	Priority_CODE	N	Mean Rank	Sum of Ranks
Technical_Debt	1.00	65	83.20	5408.00
	2.00	78	62.67	4888.00
	Total	143		

Test Statistics[a]

	Technical_Debt
Mann-Whitney U	1807.000
Wilcoxon W	4888.000
Z	-2.952
Asymp. Sig. (2-tailed)	.003

a. Grouping Variable: Priority_CODE

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Technical_Debt is the same across categories of Priority_CODE.	Independent-Samples Mann-Whitney U Test	.003	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 47 - Mann-Whitney test (Technical debt) on priority (1-Normal, 2-Urgent)

6.10.6 H6 – Response time

As seen in Figure 48, the response time that a team is able to give is around 20 hours. However, there are several outliers. We can also state that there are requests that wait for months to initiate, this may indicate that there are requests accumulating to get started, more projects transferred than initiated.
When we group the requests by priority, we can state that the response time on urgent requests is lower than on normal requests (Figure 49), which makes sense. We can also observe that the outliers on the normal priority requests reach higher values of hours.
We can also state that the response time on urgent requests is around 20 hours and the response time for normal requests is around 30 hours. This indicates that higher priority requests might be answered faster.
We want to test the hypothesis that higher priority requests are answered faster, which means that the response time is lower.
We did a Mann-Whitney test (Figure 50) to check if the distribution of response time is the same depending on the priority, which lead us to accept the null hypothesis that there is no statistical evidence that the response time decreases for higher priority requests and reject $H6_{alt}$.

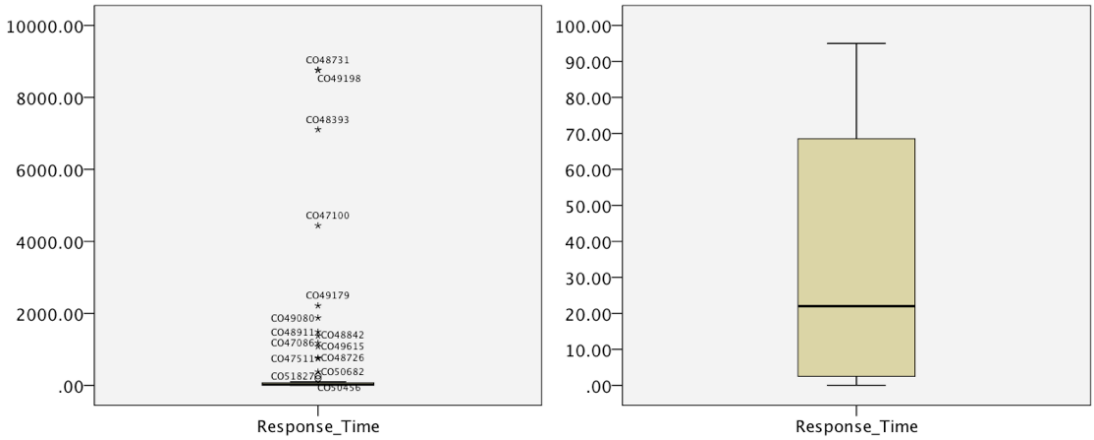

Figure 48 - Response time (hours) and close-up

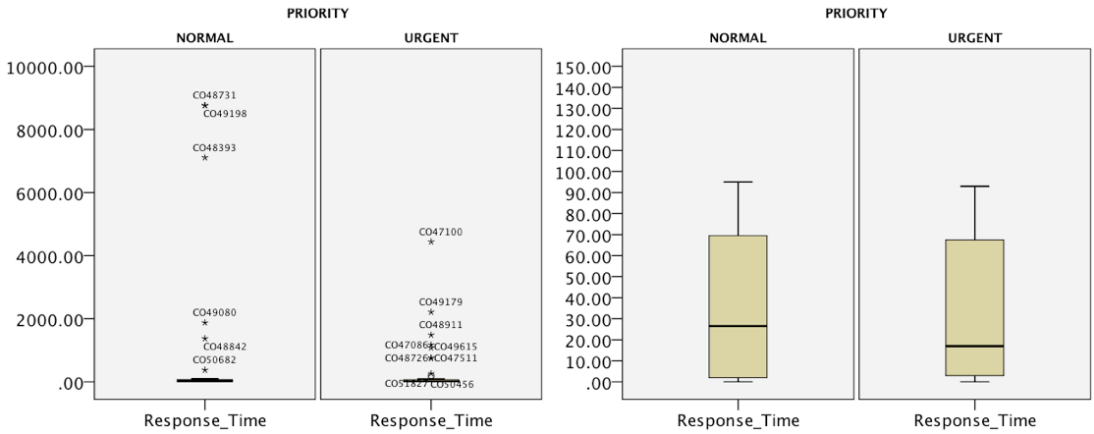

Figure 49 - Response time by priority (hours) and close-up

Mann-Whitney Test

Ranks

	Priority_CODE	N	Mean Rank	Sum of Ranks
Response_Time	1.00	116	114.75	13311.00
	2.00	103	104.65	10779.00
	Total	219		

Test Statistics[a]

	Response_Time
Mann-Whitney U	5423.000
Wilcoxon W	10779.000
Z	-1.181
Asymp. Sig. (2-tailed)	.237

a. Grouping Variable: Priority_CODE

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Response_Time is the same across categories of Priority_CODE.	Independent-Samples Mann-Whitney U Test	.237	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 50 - Mann-Whitney (Response time) on priority (1-Normal, 2-Urgent)

6.10.7 H7 – Effective time by complexity

Figure 51 shows that the effective time that a request needs to be fulfilled increases from the lowest complexity level to the highest, which indicates that more complex requests take more time to be fullfilled. However, it is interesting to see that while low complexity requests have many outliers, medium complexity requests only have one outlier and the high complexity requests do not present outliers.

We want to test the hypothesis that the complexity of the request affects the effective time that a request takes.

We did a Kruskal-Wallis test (Figure 52) to assess if the distribution of the effective time across complexity levels is the same, which lead us to reject the null hypothesis that there is no evidence that the effective time is afected by the complexity of the request and accept $H7_{alt}$.

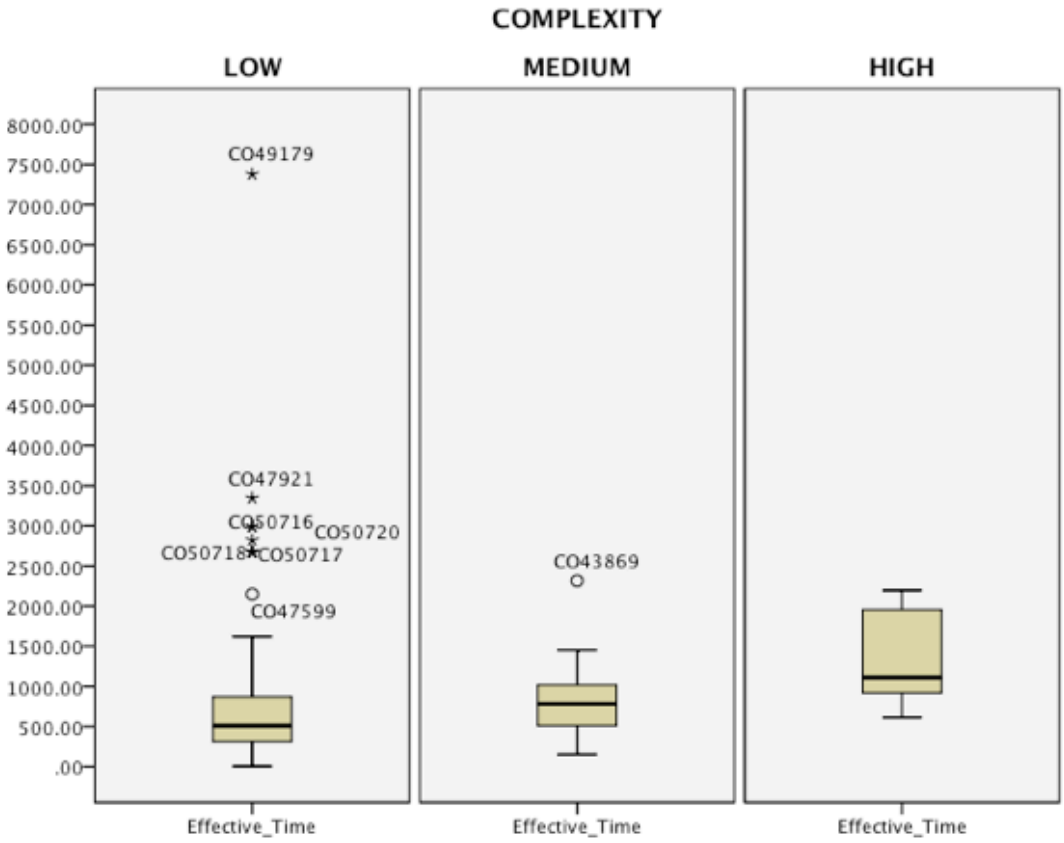

Figure 51 - Effective time by complexity (hours)

Kruskal-Wallis Test

Ranks

	Complexity_CODE	N	Mean Rank
Effective_Time	1.00	105	66.30
	2.00	29	80.95
	3.00	9	109.72
	Total	143	

Test Statistics[a,b]

	Effective_Time
Chi-Square	10.808
df	2
Asymp. Sig.	.004

a. Kruskal Wallis Test

b. Grouping Variable: Complexity_CODE

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Effective_Time is the same across categories of Complexity_CODE.	Independent-Samples Kruskal-Wallis Test	.004	Reject the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 52 - Kruskal-Wallis (Effective time) on complexity (1-Low, 2-Med, 3-High)

6.10.8 H8 – Effective time on projects containing or not a java component

In Figure 53 we identify several outliers on requests without a java component and only 3 outliers on requests with a java component. It is possible to observe that when requests have a Java component, the effective time that the request consumes is higher.

We want to test if requests with a Java component take more effective time than the others.

We did a Mann-Whitney test (Figure 54) to assess if the distribution of effective time is the same for projects that have a java component and projects that don not, which lead us to accept the null hypothesis that there is no evidence that the effective time increases for requests with a Java component and reject $H8_{alt}$.

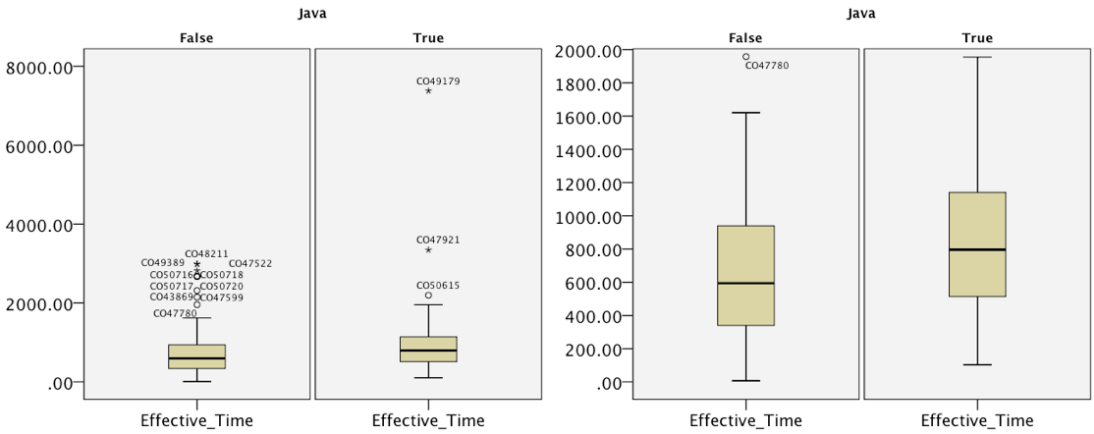

Figure 53 - Effective time by java (hours) and close-up

Mann-Whitney Test

Ranks

	Java_CODE	N	Mean Rank	Sum of Ranks
Effective_Time	.00	114	69.16	7884.00
	1.00	29	83.17	2412.00
	Total	143		

Test Statistics[a]

	Effective_Time
Mann-Whitney U	1329.000
Wilcoxon W	7884.000
Z	-1.627
Asymp. Sig. (2-tailed)	.104

a. Grouping Variable: Java_CODE

Hypothesis Test Summary

	Null Hypothesis	Test	Sig.	Decision
1	The distribution of Effective_Time is the same across categories of Java_CODE.	Independent-Samples Mann-Whitney U Test	.104	Retain the null hypothesis.

Asymptotic significances are displayed. The significance level is .05.

Figure 54 - Mann-Whitney (Effective time) on Java (0–without Java, 1–with java)

6.10.9 H9 – Forecast

Figure 55 shows us the effort line through the time and we can state that this line represents a very unstable curve which does not allow the model to produce viable forecasts using time series analysis. Most projects that were not closed at the time of the data collection were postponed to initiate on a pre-defined date (2-january-2014 9:00:00), which affected the model's curve, having a very high peek on the month January of the year 2014. It is also possible to state that the real values are higher than the estimated effort values confirming the underestimation observed previously.

We want to understand if the evolution request dataset is enables us to create forecasts based on time series analysis.

To answer RQ9, we can state that the dataset is not fit to use a forecasting model.

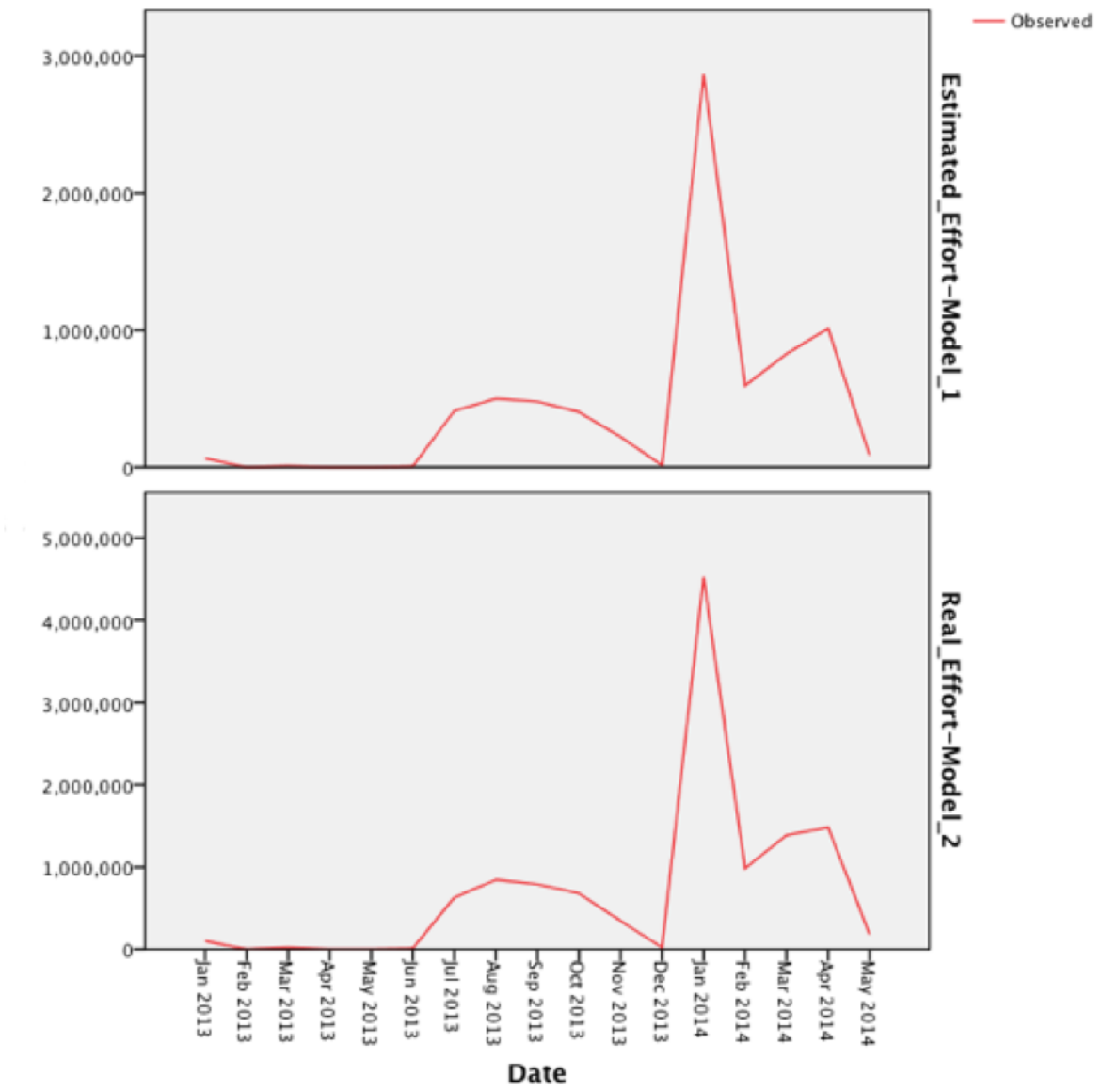

Figure 55 - Model for forecasting estimated and real effort

6.11 Inferences

6.11.1 Underestimation

When analyzing individual expert judgment, it is interesting to understand the estimators approach when estimating, as it can be either conservative/careful or risky/confident. On these requests, the estimator appears to be of the (over-)confident type. The estimator systematically underestimates the required effort to fulfill the requests.

6.11.2 Category

There are two categories very similar, the web applications category and the exclusive use tps, which could be aggregated in one category, for the sake of the analysis performed in this chapter. There are several outliers in the internal applications category, which may also indicate that this category is irregular in terms of effort that is required.

6.11.3 Complexity

The effort needed to finish a request grows with the degree of complexity as expected. However, it is interesting that low complexity requests are the ones with most outliers. This may indicate that there is more variety of low complexity requests in terms of effort, as most of them require less effort than higher complexity requests but there are also many requests that require effort values similar to the ones presented in higher complexity requests.

6.11.4 Java

Requests that are developed without a Java component present several outliers. This may indicate that these requests, despite the use of only one programming language, might need to be sub-divided to make groups more alike.

6.11.5 Technical debt

We can state that the technical debt decreases for more urgent projects, however, it is odd to see so many outliers on urgent priority requests, because urgent requests are supposed to be answered quickly (with urgency). In addition, even the most distant outlier corresponds to an urgent priority project, which means that the request is running very late, assuming that its priority decreases the technical debt as it initiates earlier than the concurrent requests that have lower priority.

6.11.6 Response time

As for the response time, we can state that the number of outliers and the hours between the median and the most distant outliers enables us to understand that the rhythm of requests for evolution arriving is faster that the rhythm of completion of the requests and some requests are left waiting for long periods of time. On the other hand, we can verify that the response time of urgent requests is lower than the response time of normal priority requests, however, the difference between these times is not long enough to be statistically significant.

6.11.7 Effective time

We can state that the effective time is higher for requests with a Java component and also for more complex requests.
The effective time shows many outliers on the low complexity requests, which may indicate that these requests may have been wrongly labeled in terms of complexity, or, despite seeming simple requests, their complexity should be calculated also taking into account the effort needed to complete the request.
As for requests with a Java component, the difference between the effective times on each type of request is not significant in a statistical level.

6.11.8 Forecasting

The SPSS tool could not produce a forecast using this evolution dataset, as more information to feed the model was required. In its current state, this requests dataset is not prone for evolution predictions using time series analysis. This may be due to a relatively small time frame available so far which prevents, for example, the detection of seasonal patterns (which would be likely to occur on a yearly basis, if they exist). The high concentration of change requests on a particular date (January, 2, 2014) also suggests that a pragmatic decision, which, from a time series analysis perspective seems to be arbitrary is masking her evolution of this time series, making it less predictable.

6.12 Threats To Validity

All the change requests originate from the same software evolution project, contracted with a particular client organization. This means that the conclusions from this particular project do not necessarily apply to other evolution projects. While the analysis approach could be applied as is, it is likely that results will vary depending on the client's profile, and perhaps even with the team within

Altran which is supporting this process (e.g. different teams may have different approaches handling the change requests, starting from the way they predict the effort required for satisfying the change requests - a more conservative / realistic estimation approach would have a dramatic impact on the technical debt, for example.
Concerning the forecasting of the required effort supporting this software evolution project in the future, using time series analysis, it would be necessary to have more evolution requests and a longer time series on the dataset to be able to produce a forecasting model.

6.13 Answer to RQ3

This assessment leads us to conclude that the estimators are underestimating the effort needed to fulfill the evolution requests. In addition, we can state by observing Figure 38 that estimate deviations are around 50% and estimators are not showing a learning pattern through time.
We stated that categories *Exclusive Use TPS* and *Web Apps* are very similar in terms of effort. We also observed that the effort demanded to complete a request increases with complexity.
The technical debt values are around 300 in general, which seems higher than desired.
In terms of effective time, we observed that the complexity of the request influences the effort needed to fulfill it, as more complex requests need more time to fulfill.
Finally, we can also state that it is necessary to have a larger project dataset in terms of period of time in order to use time series to build a forecasting model.

7 Conclusions and Future Work

7.1 Summary

The main goal of this thesis is to help Altran Portugal improve its software development estimates. To achieve this objective, we began by surveying the state of the art in order to synthetize main findings made by researchers on this area, to get the organization also informed on the possible estimation approaches that exist and to assess the current state of the practice, as this is a common concern in the corporate world. We also analyzed related studies on estimation approaches performance and gathered relevant information to help the organization to make an informed decision on the estimation approach to adopt.
Furthermore, we made an assessment to understand which were the estimation approaches that could be used in Altran Portugal. This assessment consisted in the analysis of the internal project dataset to search for input information to feed the estimation approaches and the volume of data. This analysis on the internal projects dataset allowed checking what information Altran Portugal already possesses and identifying what information is missing to increase the estimating options available. As the volume of data was not big enough to support estimation models, we focused in a type of analysis that will help expert judgment estimators to make estimates supported by past information, which is a requisite on the CMMI level 3 requirements, a level that the organization aims to achieve.
In addition, we also made a similar assessment on evolution projects that possessed a higher volume of data and tried to use this information to understand how Altran Portugal is doing in terms of client satisfaction, using metrics such as technical debt and response time.
A deeper assessment made posteriorly enabled us to conclude that estimators often transfer effort between phases, the acknowledgment of the range of estimation deviations and to find patterns when aggregating projects by programming environment, business area and size.
The analysis made on the evolution dataset showed that the estimators were underestimating the effort needed to fulfill change requests. The technical debt and response time decrease when projects have higher priority and the effective time is higher for more complex projects and for projects that present a Java component.

7.2 Impact

This thesis will raise awareness of how estimates are being performed, the estimation patterns and errors that happened in the past projects and will also provide information on previous projects necessary to produce estimates based on past projects with similar characteristics. We created a mechanism that we used to make this assessment, which will help the estimator to find information relevant to estimate projects and that should be fed with new projects to increase the volume of data available. The volume of data will have a positive impact on the information existent on past project data, increasing the probability of obtaining more accurate information and patterns. It is also important to add more projects to the database in order to increase the estimation options available, in case Altran Portugal decides to change or test another estimation approach in the future, such as a model-based approach to estimation.
The assessment on the evolution project database will enable the company to acknowledge the level of satisfaction that it brings to its clients, to state the technical debt and response time and define new goals considering these metrics. It is also relevant to make estimators understand what is the effective time needed for a project to be fulfilled and the principal factors that influence metrics such as technical debt, response time and effective time.
Both assessments enabled us to realize that estimators in Altran Portugal underestimate the effort needed to develop software development projects. The purpose of this thesis is also to inform estimators, so that the acknowledgment of the estimation errors, the systematic effort underestimation

and the estimation patterns observed lead to a improvement of the estimates of each individual through a self-assessment on their behalf, which could take the information obtained has a lessons-learned mechanism.
The fact that Altran Portugal is interested in the systematic improvement on this matter will also show its clients that despite gathering conditions to satisfy its clients, the organization strives for the excellence and gives a perfect example of how to grow through the improvement of processes and methodologies.
As Altran Portugal wants to achieve the certification CMMI level 3, this thesis also aimed to make a contribution on this goal, so, as this certification level demands that an estimator makes an estimate using past projects information, we used the information on each project, aggregated them and created a framework so that estimators could obtain such type of information.

7.3 Future Work

The future work we propose is to make a similar assessment by the time the internal project dataset reveals a volume of projects that would enable to make a statistically more relevant analysis and compare the results obtained with the actual results. It would be also important to refine and standardize the estimation mechanism that Altran Portugal actually uses, as it is actually a flaw that the estimators identified.
Moreover, with a rich project database, it could be advantageous to try to use different estimation approaches to analyze what would be the best options when estimating software development projects.
An assessment on the evolution project dataset would also be relevant in order to visualize what happened in terms of effort, technical debt, effective time and response time to demonstrate how Altran Portugal improved through time, what changes were made to better the client satisfaction and what would be the possible new improvements to be made.
It would be also very interesting to make a forecasting model using time series when the evolution project database presents a higher volume of projects, enabling the researcher to present a forecast of metrics such as effort to make predictions of future effort values on a determined date.
In both cases, it would be interesting to assess the impact of the increased visibility of the challenge the organization is facing, with respect to its estimation performance. As noted earlier, we identified a tendency for effort underestimation, both in development and in evolution projects. The increased awareness to this problem should trigger a reaction by the estimators, to progressively improve the accuracy of their predictions.

Appendix

Internal projects normality tests

Tests of Normality

	Phase = "Dev" (FILTER)	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Estimated_Effort	Selected	.203	12	.185	.818	12	.015
Real_Effort	Selected	.176	12	.200*	.919	12	.281

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figure A. 1 - Real and Estimated effort for development

Tests of Normality

	Phase = "AandD" (FILTER)	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Estimated_Effort	Selected	.172	12	.200*	.882	12	.094
Real_Effort	Selected	.232	12	.074	.851	12	.038

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figure A. 2 - Real and Estimated effort for Analysis and Design

Tests of Normality

	Phase = "Prod" (FILTER)	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Estimated_Effort	Selected	.342	12	.000	.595	12	.000
Real_Effort	Selected	.307	12	.003	.736	12	.002

a. Lilliefors Significance Correction

Figure A. 3 - Real and Estimated effort for Production

Tests of Normality

	(Project = "Proj1" | Project = "Proj2" | Project = "Proj8" | Project = "Proj10") & Phase = "AandD" (FILTER)	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Estimated_Effort	Selected	.267	4	.	.897	4	.418
Real_Effort	Selected	.305	4	.	.788	4	.083

a. Lilliefors Significance Correction

Figure A. 4 - Real and Estimated effort for Analysis and Design on .Net

Tests of Normality

	(Project = "Proj4" | Project = "Proj5" | Project = "Proj6" | Project = "Proj10"| Project = "Proj11" | Project = "Proj12") & Phase = "Dev" (FILTER)	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Estimated_Effort	Selected	.178	6	.200*	.951	6	.744
Real_Effort	Selected	.236	6	.200*	.954	6	.775

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figure A. 5 - Real and Estimated effort for Development on BI

Tests of Normality

	(Project = "Proj7" | Project = "Proj8") & Phase = "Dev" (FILTER)	Kolmogorov-Smirnov[a]		
		Statistic	df	Sig.
Estimated_Effort	Selected	.260	2	.
Real_Effort	Selected	.260	2	.

a. Lilliefors Significance Correction

Figure A. 6 - Real and Estimated effort for Development on Healthcare

Tests of Normality

	Size	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Error	1	.347	3	.	.836	3	.203
	2	.334	5	.071	.818	5	.113
	3	.189	4	.	.984	4	.927

a. Lilliefors Significance Correction

Figure A. 7 - Error on the size of the projects (1-Small, 2-Med, 3-Large)

Tests of Normality[a]

	WBS	Kolmogorov-Smirnov[b]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Error	3	.338	5	.063	.814	5	.106
	4	.260	2	.			
	5	.260	2	.			
	6	.260	2	.			

a. Error is constant when WBS = 2. It has been omitted.

b. Lilliefors Significance Correction

Figure A. 8 - Error on the level of the WBS

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Estimated_Effort	.179	12	.200*	.924	12	.321
Real_Effort	.206	12	.170	.894	12	.132

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figure A. 9 – Total Estimated and Total Real effort of all projects

Evolution requests normality tests

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Total_Effort	.225	219	.000	.705	219	.000
Estimate	.234	219	.000	.673	219	.000

a. Lilliefors Significance Correction

Figure A. 10 – Total Estimated and Real effort of all projects

Tests of Normality

		Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	CATEGORY	Statistic	df	Sig.	Statistic	df	Sig.
Total_Effort	Excl Use	.213	25	.005	.723	25	.000
	Int Apps	.260	139	.000	.596	139	.000
	Web Apps	.177	55	.000	.866	55	.000
Estimate	Excl Use	.231	25	.001	.716	25	.000
	Int Apps	.275	139	.000	.531	139	.000
	Web Apps	.169	55	.000	.865	55	.000

a. Lilliefors Significance Correction

Figure A. 11 - Estimated and Real effort on category

Tests of Normality

		Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	COMPLEXITY	Statistic	df	Sig.	Statistic	df	Sig.
Total_Effort	HIGH	.156	24	.135	.937	24	.143
	LOW	.215	147	.000	.583	147	.000
	MEDIUM	.095	48	.200*	.915	48	.002
Estimate	HIGH	.191	24	.024	.885	24	.011
	LOW	.213	147	.000	.627	147	.000
	MEDIUM	.096	48	.200*	.917	48	.002

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figure A. 12 - Estimated and Real effort on complexity

Tests of Normality

	Java_CODE	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Total_Effort	.00	.251	170	.000	.605	170	.000
	1.00	.145	49	.012	.901	49	.001
Estimate	.00	.248	170	.000	.611	170	.000
	1.00	.143	49	.014	.834	49	.000

a. Lilliefors Significance Correction

Figure A. 13 - Estimated and Real effort on Java (0–without Java, 1–with java)

Tests of Normality

	Java_CODE	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Technical_Debt	.00	.227	114	.000	.711	114	.000
	1.00	.300	29	.000	.512	29	.000

a. Lilliefors Significance Correction

Figure A. 14 - Technical debt on Java (0–without Java, 1–with java)

Tests of Normality

	Priority_CODE	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Response_Time	1.00	.502	116	.000	.194	116	.000
	2.00	.450	103	.000	.274	103	.000

a. Lilliefors Significance Correction

Figure A. 15 - Response time on priority (1-Normal, 2-Urgent)

Tests of Normality

	Complexity_CODE	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Effective_Time	1.00	.238	105	.000	.619	105	.000
	2.00	.139	29	.162	.922	29	.034
	3.00	.199	9	.200*	.909	9	.306

*. This is a lower bound of the true significance.

a. Lilliefors Significance Correction

Figure A. 16 - Effective time on complexity (1–Low, 2–Med, 3–High)

Tests of Normality

	Java_CODE	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Effective_Time	.00	.193	114	.000	.789	114	.000
	1.00	.288	29	.000	.590	29	.000

a. Lilliefors Significance Correction

Figure A. 17 - Effective time on Java component (0–without Java, 1–with java)

Bibliography

[1] I. Sommerville, *Software engineering (7th edition)*. 2004, p. 784.

[2] Pmi, *A Guide to the Project Management Body of Knowledge*, vol. 1, no. 11. Project Management Institute, 2008, p. 459.

[3] S. McConnell, *Software Estimation: Demystifying the Black Art*. Redmond, WA, USA: Microsoft Press, 2006.

[4] J. C. Cunha, S. Cruz, M. Costa, A. R. Rodrigues, and M. Vieira, "Implementing Software Effort Estimation in a Medium-sized Company," *2011 IEEE 34th Softw. Eng. Work.*, pp. 92–96, Jun. 2011.

[5] T. Menzies and J. Hihn, "Evidence-based cost estimation better-quality for software," *Software, IEEE*, pp. 64–66, 2006.

[6] R. Fairley, *MANAGING AND LEADING SOFTWARE PROJECTS*. A JOHN WILEY & SONS, INC., 2009, p. 492.

[7] R. S. Pressman, *Software engineePressman, R. S. (n.d.). Software engineering (2nd ed.). New York: McGraw-Hill Book Company.ring*, 2nd ed. New York: McGraw-Hill Book Company.

[8] S. Fingerman, "Practical software project estimation; a toolkit for estimating software development effort & duration," *Sci-Tech News*, vol. 65, no. 1, p. 28, 2011.

[9] B. W. Boehm, "Software Engineering Economics," *IEEE Trans. Softw. Eng.*, vol. SE-10, no. 1, pp. 4–21, Jan. 1984.

[10] "CMMI." [Online]. Available: http://cmmiinstitute.com. [Accessed: 15-Oct-2013].

[11] M. Jorgensen and M. Shepperd, "A Systematic Review of Software Development Cost Estimation Studies," *IEEE Transactions on Software Engineering*, vol. 33. pp. 33–53, 2007.

[12] M. Jørgensen, "Forecasting of software development work effort: Evidence on expert judgement and formal models," *Int. J. Forecast.*, 2007.

[13] B. Lo and X. Gao, "Assessing software cost estimation models: criteria for accuracy, consistency and regression," *Aust. J. Inf. Syst.*, no. 1, pp. 30–44, 1997.

[14] S. Grimstad and M. Jørgensen, "A framework for the analysis of software cost estimation accuracy," *Proc. 2006 ACM/IEEE Int. Symp. Int. Symp. Empir. Softw. Eng. - ISESE '06*, p. 58, 2006.

[15] L. C. Briand and I. Wieczorek, “Resource Estimation in Software Engineering,” *Engineering*. pp. 1–67, 2001.

[16] K. Moløkken-østvold and P. O. Box, “Combining Estimates with Planning Poker – An Empirical Study Nils Christian Haugen,” *Aust. Softw. Eng. Conf.*, 2007.

[17] F. J. Heemstra, “Software cost estimation,” *Inf. Softw. Technol.*, vol. 34, no. 10, pp. 627–639, Oct. 1992.

[18] Chaos, “The standish group report,” *Chaos*, vol. 49, pp. 1–8, 1995.

[19] H. Leung and Z. Fan, “Software Cost Estimation,” pp. 1–14, 2002.

[20] M. Jørgensen, “A review of studies on expert estimation of software development effort,” *J. Syst. Softw.*, vol. 70, no. 1–2, pp. 37–60, Feb. 2004.

[21] K. Moløkken-Østvold and M. Jørgensen, “Group Processes in Software Effort Estimation,” *Empir. Softw. Eng.*, vol. 9, no. 4, pp. 315–334, Dec. 2004.

[22] A. L. Lederer and J. Prasad, “Nine management guidelines for better cost estimating,” *Commun. ACM*, vol. 35, 1992.

[23] N. Harvey, “Improving judgment in forecasting,” in *INTERNATIONAL SERIES IN OPERATIONS RESEARCH AND MANAGEMENT SCIENCE*, 2001, pp. 59–80.

[24] S. S. Vicinanza, T. Mukhopadhyay, and M. J. Prietula, “Software-Effort Estimation: An Exploratory Study of Expert Performance,” *Information Systems Research*, vol. 2. pp. 243–262, 1991.

[25] I. Myrtveit and E. Stensrud, “A controlled experiment to assess the benefits of estimating with analogy and regression models,” *IEEE Trans. Softw. Eng.*, vol. 25, 1999.

[26] R. T. Hughes, “Expert judgement as an estimating method,” *Inf. Softw. Technol.*, vol. 38, no. 2, pp. 67–75, Jan. 1996.

[27] N. C. Dalkley, “Delphi,” *Second Symp. Long-Range Forecast. Plan.*, 1967.

[28] G. Rowe and G. Wright, “The Delphi technique as a forecasting tool: issues and analysis,” *Int. J. Forecast.*, vol. 15, no. 4, pp. 353–375, Oct. 1999.

[29] K. Wiegers, “Stop promising miracles,” *Softw. Dev.*, no. February, pp. 1–8, 2000.

[30] C. Rush and R. Roy, “Expert Judgement in Cost Estimating: Modelling the Reasoning Process,” *Concurr. Eng.*, vol. 9, no. 4, pp. 271–284, Dec. 2001.

[31] J. Grenning, “Planning poker or how to avoid analysis paralysis while release planning,” *Hawthorn Woods Renaiss. Softw.*, no. April, pp. 1–3, 2002.

[32] B. A. Kitchenham, E. Mendes, and G. H. Travassos, “Cross- vs. Within-Company Cost Estimation Studies: A Systematic Review,” *IEEE Trans. Softw. Eng.*, pp. 316–329, 2007.

[33] L. C. Briand, T. Langley, and I. Wieczorek, “A replicated assessment and comparison of common software cost modeling techniques,” *Int. Conf. Softw. Eng.*, p. 377, 2000.

[34] L. C. Briand, K. El Emam, D. Surmann, I. Wieczorek, and K. D. Maxwell, “An assessment and comparison of common software cost estimation modeling techniques,” *Proc. 21st Int. Conf. Softw. Eng. - ICSE ’99*, pp. 313–322, 1999.

[35] B. Boehm, C. Abts, and S. Chulani, “Software development cost estimation approaches—A survey,” *Ann. Softw. Eng.*, vol. 10, pp. 177–205, 2000.

[36] “Price.” [Online]. Available: http://www.pricesystems.com. [Accessed: 02-Nov-2013].

[37] R. W. Zmud and C. F. Kemerer, “An Empirical Validation of Software Cost Estimation Models,” vol. 30, no. 5, 1987.

[38] L. C. Briand, V. R. Basili, and W. M. Thomas, “A pattern recognition approach for software engineering data analysis,” *Softw. Eng. IEEE Trans.*, vol. 18, pp. 931–942, 1992.

[39] L. C. Briand, V. R. Basili, and C. J. Hetmanski, “Developing interpretable models with optimized set reduction for identifying high-risk software components,” *IEEE Trans. Softw. Eng.*, vol. 19, pp. 1028–1044, 1993.

[40] L. C. Briand, K. El Emam, and F. Bomarius, “COBRA: a hybrid method for software cost estimation, benchmarking, and risk assessment,” *Proc. 20th Int. Conf. Softw. Eng.*, 1998.

[41] A. Rubin, “MEASURING SOFTWARE ITS IMPACT PROCESS MATURITY : ON PRODUCTIVITY,” pp. 468–476, 1990.

[42] A. Trendowicz, J. Heidrich, and J. Münch, “Development of a hybrid cost estimation model in an iterative manner,” p. 331, 2006.

[43] O. Dieste and N. Juristo, “Systematic Review and Aggregation of Empirical Studies on Elicitation Techniques,” *IEEE Trans. Softw. Eng.*, vol. 37, pp. 283–304, 2011.

[44] R. Jeffery, M. Ruhe, and I. Wieczorek, “A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data,” *Inf. Softw. Technol.*, 2000.

[45] “ISBSG.” [Online]. Available: http://www.isbsg.org. [Accessed: 21-Dec-2013].